Model-Based and Data-Driven Methodologies Toward Predictive Digital Twins of Structures

Doctoral Dissertation of

Matteo Torzoni

Department of Civil and Environmental Engineering

Doctoral Programme in
Structural, Seismic and Geotechnical Engineering

In Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in
Structural, Seismic and Geotechnical Engineering

Politecnico di Milano

May 2024 – XXXVI cycle

© Politecnico di Milano. All rights reserved.

Doctoral Dissertation of
Matteo Torzoni

Thesis Supervisor
Prof. Stefano Mariani

Thesis Supervisor
Prof. Andrea Manzoni

Thesis Tutor
Prof. Fabio Biondini

Chair of the Doctoral Programme
Prof. Stefano Mariani

POLITECNICO
MILANO 1863
DIPARTIMENTO DI
INGEGNERIA CIVILE E AMBIENTALE

This page intentionally left blank.

Model-Based and Data-Driven Methodologies Toward Predictive Digital Twins of Structures

Doctoral Dissertation of
Matteo Torzoni

Submitted to the Department of
Civil and Environmental Engineering on May, 2024,
in Partial Fulfillment of the Requirements for the
Degree Doctor of Philosophy in
Structural, Seismic and Geotechnical Engineering

ABSTRACT

The digital twin concept has emerged as a paradigm to enable diagnostic and predictive capabilities that are not achievable with computational models alone. A digital twin is a personalized virtual representation of a physical asset that evolves over time. It relies on a set of computational models that are dynamically updated to persistently mirror the physical counterpart throughout its life cycle. Additionally, it features predictive capabilities that inform decisions tailored to realize value in the physical setting of interest. A bi-directional interaction between the physical and virtual domains, comprising either automated or human-in-the-loop feedback flows, is a critical enabler for digital twins. Enhanced computational efficiency is also required to handle the continuous assimilation of noisy and big data, as well as to quantify and propagate relevant uncertainties.

This Thesis aims to advance digital twin technologies for monitoring the structural integrity of engineering systems by hybridizing physics-based modeling with artificial intelligence. Enabling a digital twin perspective for critical structural systems, whether for safety or operational reasons, can unlock the potential of condition-based and predictive maintenance practices, yielding numerous benefits throughout their life cycle.

We begin by laying down a theoretical groundwork and integrating cutting-edge insights from data science and engineering. This encompasses: (*i*) model-based and data-driven approaches to parameter identification; (*ii*) full and reduced-order computational models in structural dynamics to formulate damage identification strategies within the simulation-based

paradigm of structural health monitoring; and (*iii*) machine learning techniques to extract information from sensor data, forecast future system responses, and determine optimal courses of action. Subsequently, we propose four contributions that combine simulation-based supervision, essential for addressing the lack of experimental data related to damage onset and propagation, with learnable mappings aimed at assimilating vibration recordings, enhancing damage identification capabilities, and improving computational efficiency. Firstly, we introduce a data-driven diagnostic framework that leverages deep metric learning to map raw sensor recordings onto a damage-sensitive low-dimensional space. We investigate its ability to locate damage by exploring different training options within a Siamese architecture, demonstrating remarkable accuracy and robustness against variations in the extent and severity of damage. Secondly, we tackle the computational efficiency of surrogate models for structural health monitoring. We employ both full and reduced-order physics-based models to sequentially train surrogate components with increasing levels of approximation. The resulting multi-fidelity surrogates serve both model-updating purposes and the generation of large datasets, demonstrating superior accuracy compared to single-fidelity approximations without increasing computational costs. As a third contribution, we present a hybrid data/model approach for real-time damage identification, enhancing Bayesian model updating through learnable mappings. Damage-sensitive features are extracted from sensor recordings via deep metric learning, and then exploited within a Markov chain Monte Carlo sampler to assess damage with quantified uncertainty. The achieved estimation accuracy and computational efficiency overcome limitations linked to the slow convergence of sampling algorithms and the curse of dimensionality affecting the inference of low-sensitivity damage parameters. Finally, we propose a digital twin framework relying on a probabilistic graphical model to encode the asset-twin interaction via observational data and control inputs. The decision-making capabilities of health-aware digital twins are assessed through demonstrations of data-driven health monitoring, prediction, and planning. While experimental data are not used, all methodologies are assessed across various case studies using noisy synthetic measurements.

This Thesis is dedicated to my parents,

to whom all my love goes.

This page intentionally left blank.

PhD School in Structural, Seismic and Geotechnical Engineering
Department of Civil and Environmental Engineering
Politecnico di Milano – XXXVI cycle

Faculty Board

Prof. Stefano Mariani (Coordinator)
Prof. Garbeiel Della Vecchia (Vice-Coordinator)
Prof. Dario Coronelli (Vice-Coordinator)
Prof. Raffaele Ardito
Prof. Patrick Bamonte
Prof. Fabio Biondini
Prof. Gabriella Bolzon
Prof. Matteo Bruggi
Prof. Claudia Comi
Prof. Alberto Corigliano
Prof. Massimiliano Cremonesi
Prof. Claudio Di Prisco
Prof. Marco Di Prisco
Prof. Roberto Felicetti
Prof. Liberato Ferrara
Prof. Attilio Alberto Frangi
Prof. Aldo Ghisi
Prof. Cristina Jommi
Prof. Maurizio Lualdi
Prof. Luca Martinelli
Prof. Giovanni Muciaccia
Prof. Roberto Paolucci
Prof. Umberto Perego
Prof. Lorenza Maria Petrini
Prof. Chiara Smerzini
Prof. Claudio Tamagnini

This page intentionally left blank.

ACKNOWLEDGEMENTS

I owe immeasurable gratitude to Prof. Stefano Mariani and Prof. Andrea Manzoni for their time, resources, and invaluable guidance during my PhD studies, always allowing me to pursue my own inclinations. I am thankful for the opportunity to learn from you. Your vision, passion, enthusiasm, and empathy have been crucial for this Thesis. The many pieces of advice and words of encouragement you gave me in the face of difficulties are treasures I will carry with me forever.

I am also extremely grateful to Prof. Karen Willcox for her mentoring and for giving me the opportunity to spend five months at the Oden Institute for Computational Engineering and Sciences. My time in Austin was an incredibly enriching experience that taught me a great deal both in research and in life. Thanks also to Dr. Marco Tezzele for being a great mentor and a sincere friend away from home.

I would like to acknowledge Prof. Alberto Corigliano for continuing to foster my growth since my Master Thesis. Our discussions and his broad vision have always sparked stimulating perspectives. Thanks to Dr. Luca Rosafalco for introducing me into the world of structural health monitoring and machine learning, as well as for being a dear friend. Thanks to the crew of PhD students (past and present) for the spontaneous friendship we have built. Thanks to Daniel Calegaro for being a perfect desk mate. Thanks to Paolo Conti for being like a second family while I was in the US. Thanks to all the PhD students, researchers, and professors who have consciously or unconsciously contributed to my academic growth during these years.

Thank you to my lifelong friends from Arezzo for always being there — you are my anchor. Thank you to Alice for sharing with me the joys and fears of the last few years, and for your support, understanding, and affection. Finally, I would like to thank my family. This journey has been possible only because of your constant support, and I am grateful for it — so far away, yet always so close.

Matteo Torzoni
Milano, May 2024

This page intentionally left blank.

CONTENTS

Abstract V

Nomenclature XVII

Acronyms XXIX

List of Figures XXXI

List of Tables XXXIII

1 Introduction 1
- 1.1 Why Predictive Digital Twins of Structures? 1
- 1.2 The Need for Structural Health Monitoring 5
  - 1.2.1 Simulation-Based Paradigm 7
  - 1.2.2 Model-Based and Data-Driven Approaches 9
  - 1.2.3 Operational Evaluation 14
  - 1.2.4 Coping with the Lack of Data 16
- 1.3 The Need for Reduced-Order and Surrogate Models 17
  - 1.3.1 Many-Query Applications 17
  - 1.3.2 Multi-Fidelity Methods 18
- 1.4 Thesis Objectives 19
- 1.5 Thesis Outline 26

2 Model-Based and Data-Driven Approaches to Parameter Identification 31
- 2.1 A Model-Based Approach: MCMC sampling 31
  - 2.1.1 Parameter Estimation in a Bayesian Framework 31
  - 2.1.2 Metropolis and Metropolis-Hastings Samplers 35
- 2.2 A Data-Driven Approach: k-Nearest Neighbors 40
- 2.3 Thesis Perspective 42

3 Numerical Models for the Simulation-Based Paradigm 45
- 3.1 Full-Order Model 46
- 3.2 Damage and Operational Conditions 47
- 3.3 Reduced-Order Modeling 49
  - 3.3.1 Proper Orthogonal Decomposition 49
  - 3.3.2 Reduced Basis Method in Elasto-Dynamics 53
- 3.4 Thesis Perspective 56

4 Machine Learning for Predictive Digital Twins 59
4.1 Key Aspects of Machine Learning 60
4.1.1 Regression Perspective 60
4.1.2 Classification Perspective 62
4.1.3 Quality and Quantity of Training Data 64
4.1.4 Validation and Testing 64
4.1.5 Deep Learning 65
4.2 Neural Networks 65
4.2.1 Fully-Connected Layer 66
4.2.2 Last Layer Activation Function 67
4.2.3 Optimization for Training Deep Models 69
4.2.4 Regularization for Deep Learning Models 72
4.2.5 1D Convolutional Layer 75
4.2.6 Long Short-Term Memory Layer 78
4.2.7 Autoencoders 80
4.3 Probabilistic Graphical Models 83
4.3.1 Graph Formalism 84
4.3.2 Bayesian Networks 85
4.3.3 Dynamic Bayesian Networks 87
4.3.4 Utility and Sequential Decision Making 90
4.3.5 Dynamic Decision Networks 94
4.4 Thesis Perspective 97

5 Deep Metric Learning for Damage Localization 101
5.1 Deep Metric Learning 102
5.1.1 Data Pairing Process 103
5.1.2 Siamese Architecture 104
5.1.3 pairwise Loss Functions 105
5.1.4 Energy-Based Models Perspective 106
5.2 Regression in the Low-Dimensional Metric Space 107
5.3 Damage Localization: L-Shaped Cantilever Beam 108
5.3.1 Dataset Assembly and Training Options 109
5.3.2 Localization Outcome: Numerical Results 110
5.4 Damage Localization: Railway Bridge 122
5.4.1 Dataset Assembly and Training Options 122
5.4.2 Localization Outcome: Numerical Results 128
5.5 Conclusions 132

6 Multi-Fidelity Surrogate Modeling 135
6.1 Multi-Fidelity Deep Neural Net Surrogate Model 136
6.1.1 Low/High Fidelity Datasets 137
6.1.2 Low/High Fidelity Surrogate Components 139
6.2 Validation and Verification of Surrogate Modeling 142
6.2.1 Surrogate of an L-Shaped Cantilever Beam 143
6.2.2 Surrogate of a Railway Bridge 147
6.3 Simulated Identification: L-Shaped Beam 152

6.3.1 Data-Driven Preliminary Identification 153
6.3.2 Markov Chain Monte Carlo: Simulation Setup 156
6.3.3 Parameter Identification Outcome 159
6.4 Conclusions 161

7 Enhancing Bayesian Model Updating via Deep Nets 165
7.1 Feature-Based Markov Chain Monte Carlo Sampling 167
7.1.1 Feature Extractor Model 169
7.1.2 Feature-Oriented Surrogate Model 170
7.1.3 Markov Chain Monte Carlo: Simulation Setup 172
7.2 Parameter Identification: Numerical Experiments 174
7.2.1 L-Shaped Cantilever Beam 174
7.2.2 Portal Frame 178
7.2.3 Railway Bridge 180
7.3 Conclusions 183

8 Digital Twins for Optimal Management and Maintenance of Structures via Probabilistic Graphical Models 185
8.1 Digital Twin Concept for Engineering Structures 187
8.1.1 Mathematical Encoding of Asset-Twin Systems 187
8.1.2 Extension to Prediction and Optimal Planning 191
8.2 Data Assimilation via Artificial Neural Networks 194
8.2.1 Damage Identification 194
8.2.2 Digital State Identification 196
8.3 Demonstration: L-Shaped Cantilever Beam 197
8.3.1 Dataset Assembly 197
8.3.2 Digital Twin framework 197
8.3.3 Results: Two Available Actions 199
8.3.4 Results: Four Available Actions 203
8.4 Demonstration: Railway Bridge 206
8.4.1 Dataset Assembly 206
8.4.2 Digital Twin Framework 206
8.4.3 Results 209
8.5 Conclusions 211

9 Conclusions and Outlook 213
9.1 Summary of Contributions 213
9.2 Limitations 216
9.3 Opportunities for Further Research 218

Bibliography 221

Appendices
A Implementation details of $\mathrm{NN}_{\mathbf{h}}$ 263
B Implementation details of $\mathrm{NN}_{\mathrm{MF}}$ and $\mathrm{NN}_{\mathrm{OC}}$ 265
C Imaging Time Series: Markov Transition Field 269

D Implementation details of NN_{CAE} and NN_{SUR} 271
E Implementation details of NN_{CL} and NN_{RG} 273

NOMENCLATURE

$\mathbf{A}$	Matrix of reals
$\mathbf{A}_{\mathrm{RB}}$	Low-rank approximation of $\mathbf{A}$
$\mathcal{A}^{n_S}$	Space activation function for the n_S-th sleeper
$\mathcal{A}^{n_S}_{n_A}$	Time-modulating function related to the n_A-th axle for the n_S-th sleeper
$\boldsymbol{\mathscr{A}}$	Tensor of scalar values
$\widetilde{\boldsymbol{\mathscr{A}}}^{\ell}$	Set of convolution feature maps obtained from C_ℓ
$\mathbf{a}_{n_C}$	n_C-th column of $\mathbf{A}$
$\boldsymbol{a}$	Vector of scalar values
$\widetilde{\boldsymbol{a}}^{\ell}$	Convolution feature map obtained from C_ℓ
a	Scalar value
$\mathbb{B}$	Space of Boolean numbers
$\mathbf{B}$	$N^{\mathbf{A}}_{\mathrm{RB}}$-dimensional orthogonal basis in $\mathbb{R}^{N^{\mathbf{A}}_R}$
B	Mini-batch size
$\mathbf{b}$	One-hot encoding of ℓ
$\widehat{\mathbf{b}}$	Data-driven prediction of $\mathbf{b}$
ℓ	Damage location among a set of predefined damage scenarios
$\mathbf{C}$	Damping matrix
$\mathbf{C}_r$	Reduced damping matrix
C_s	Cosine similarity
C_ℓ	ℓ-th one-dimensional convolutional layer
$\underline{\mathrm{Ch}}(X_{N_i})$	Children nodes of X_{N_i}
$\mathbf{c}^{\ell}$	Cell state of $\mathcal{R}^{\ell}$
$\widetilde{\mathbf{c}}^{\ell}$	New candidate cell state in the input gate of $\mathcal{R}^{\ell}$
c	Normalization constant
$\mathcal{D}$	Training dataset
$\mathcal{D}_{\mathrm{AE}}$	Training dataset for an autoencoder model
$\mathcal{D}_{\mathrm{CL}}$	Training dataset for a classification model

$\mathcal{D}_{\mathrm{CL}}^{\mathbf{h}}$	Feature encoding of $\mathcal{D}_{\mathrm{CL}}$
$\mathcal{D}_{\mathrm{DT}}$	Digital twin training dataset
$\mathcal{D}_{\mathrm{HF}}$	High-fidelity training dataset
$\mathcal{D}_{\mathrm{HF}}^{\mathrm{OC}}$	Feature encoding of $\mathcal{D}_{\mathrm{HF}}$ via $\mathrm{NN}_{\mathrm{OC}}$
$\mathcal{D}_{\mathcal{I}}$	MTF encoding of $\mathcal{D}_{\mathrm{MF}}$
$\mathcal{D}_{\mathcal{I}}^{\mathrm{P}}$	Version of $\mathcal{D}_{\mathcal{I}}$ augmented via data pairing process
$\mathcal{D}_{\mathrm{LF}}$	Low-fidelity training dataset
$\mathcal{D}_{\mathrm{LF}}^{\mathsf{vec}}$	Vectorized low-fidelity training dataset
$\mathcal{D}_{\mathrm{LOC}}$	Training dataset for a damage localization model
$\mathcal{D}_{\mathrm{LOC}}^{\mathbf{h}}$	Feature encoding of $\mathcal{D}_{\mathrm{LOC}}$ via $\mathrm{NN}_{\mathbf{h}}$
$\mathcal{D}_{\mathrm{LOC}}^{\mathrm{P}}$	Version of $\mathcal{D}_{\mathrm{LOC}}$ augmented via data pairing process
$\mathcal{D}_{\mathrm{MF}}$	Training dataset generated through $\mathrm{NN}_{\mathrm{MF}}$
$\mathcal{D}_{\mathrm{OC}}^{\mathrm{P}}$	Version of $\mathcal{D}_{\mathrm{HF}}$ augmented via data pairing process
$\mathcal{D}_{\mathrm{RG}}^{\mathbf{h}}$	Feature encoding of $\mathcal{D}_{\mathrm{RG}}$
$\mathcal{D}_{\mathrm{RG}}$	Training dataset for a regression model
$\mathcal{D}_{\mathrm{SM}}$	Training dataset for a surrogate model
$\mathcal{D}_{\boldsymbol{\theta}}^{\mathbf{h}}$	Feature encoding of $\mathcal{D}_{\mathcal{I}}$ via $\mathrm{NN}_{\mathrm{ENC}}$
d	Digital state
d^{NN}	Assimilated digital state
$\mathbb{E}$	Expectation operator
$\mathbb{E}_{\pi}$	Expectation under policy π
E	Young's modulus
$\mathcal{E}_{\mathbf{h}}$	Distance function of feature representations
$\mathcal{E}_{\mathrm{LF}}$	Euclidean distance between two operational conditions
$\mathcal{E}_{\mathbf{y}}$	Euclidean distance between two damage positions
$\overline{\mathcal{E}_{\mathbf{y}}}$	Threshold distance characterizing the similarity of data in terms of $\mathbf{y}$
$\overline{\mathcal{E}_{\boldsymbol{\theta}}}$	Threshold distance characterizing the similarity of data in terms of $\boldsymbol{\theta}$
$\mathscr{E}$	Set of edges
$\mathbf{e}$	Direction unit vector
F_A	Force associated with the axle-carried mass m_A

$\mathcal{F}^{\ell}$	ℓ-th fully-connected layer
$\widetilde{\mathcal{F}}$	Target fully-connected mapping
$\mathbf{f}$	Nodal load vector
$\mathbf{f}_r$	Reduced load vector
f_q	Load frequency of $q(t)$
$\widehat{f}_q$	Data-driven prediction of f_q
f_{s}	Sampling frequency for $\mathbf{U}$
$\mathcal{G}$	Graph data structure
$\mathcal{G}^0$	Bayesian network over the initial states distribution of a dynamic Bayesian network
$\mathcal{G}^{\rightarrow}$	Two-time-slice Bayesian network for the process of a dynamic Bayesian network
$H_{\mathcal{C}}^{\ell}$	Size of the filter kernels in $\mathcal{C}_{\ell}$
$H_{\mathcal{P}}^{\ell}$	Pool size of $\mathcal{P}_{\max}^{\ell}$
$\mathcal{H}$	Categorical cross-entropy
$\mathscr{H}$	Heavyside function
$\mathbf{h}$	Training data feature
$\widehat{\mathbf{h}}$	Approximation of $\mathbf{h}$ via forward model
$\mathbf{h}^{\mathrm{EXP}}$	Query data feature
$\mathbf{h}_{1,\ldots,N_{\mathrm{obs}}}^{\mathrm{EXP}}$	Batch of N_{obs} query data features
$\mathbf{h}_{\varepsilon}$	Discrepancy term
$\mathbf{I}_{\mathrm{RB}}$	$N_{\mathrm{RB}}^{\mathbf{A}}$-dimensional identity matrix
$\mathcal{I}$	Grey-scale mosaic of MTF encodings
$\widehat{\mathcal{I}}$	Data-driven prediction of $\mathcal{I}$
$\mathbf{K}$	Stiffness matrix
$\mathbf{K}_r$	Reduced stiffness matrix
k_{robin}	Elastic coefficient associated with a Robin mixed boundary condition
L	Number of measurements in each time series of sensor recordings
L_{C}	Chunk length
L_{chain}	Markov chain length
L_{LF}	Number of POD modes in $\mathbf{Y}_{\mathrm{LF}}$
L_{ℓ}	Length of the input feature representation to $\mathcal{C}_{\ell}$
L_{OC}	Number of POD modes in $\mathbf{Y}_{\mathrm{OC}}$

L_{vec}	Length of the vectorized vibration recordings instance
$\mathcal{L}$	Loss function
$\mathcal{L}^{R2}$	L^2-regularized loss function
$\mathcal{L}_{\text{AE}}$	Autoencoder loss function
$\mathcal{L}_{\text{CAE}}$	NN_{CAE} loss function
$\mathcal{L}^{R2}_{\text{CAE}}$	L^2-regularized NN_{CAE} loss function
$\mathcal{L}_{\text{CL}}$	NN_{CL} loss function
$\mathcal{L}^{R2}_{\text{CL}}$	L^2-regularized NN_{CL} loss function
$\mathcal{L}_{\text{HF}}$	NN_{HF} loss function
$\mathcal{L}^{R2}_{\text{HF}}$	L^2-regularized NN_{HF} loss function
$\mathcal{L}_{\mathbf{h}}$	$\text{NN}_{\mathbf{h}}$ pair-wise loss function
$\mathcal{L}^{R2}_{\mathbf{h}}$	L^2-regularized $\text{NN}_{\mathbf{h}}$ pair-wise loss function
$\mathcal{L}_{\text{LF}}$	NN_{LF} loss function
$\mathcal{L}^{R2}_{\text{LF}}$	L^2-regularized NN_{LF} loss function
$\mathcal{L}_{\text{OC}}$	NN_{OC} loss function
$\mathcal{L}^{R2}_{\text{OC}}$	L^2-regularized NN_{OC} loss function
$\mathcal{L}_{\text{RG}}$	NN_{RG} loss function
$\mathcal{L}^{R2}_{\text{RG}}$	L^2-regularized NN_{RG} loss function
$\mathcal{L}_{\text{SUR}}$	NN_{SUR} loss function
$\mathcal{L}^{R2}_{\text{SUR}}$	L^2-regularized NN_{SUR} loss function
$\mathbf{M}$	Mass matrix
$\mathbf{M}_r$	Reduced mass matrix
$\mathcal{M}$	Forward computational model
$\mathcal{M}_{\text{CL}}$	Classification model
$\mathcal{M}_{\text{k-NN}}$	k-NN regressor
$\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$	k-NN regressor targeting $\mathbf{y}$
$\mathcal{M}^{\boldsymbol{\mu}^{\text{LF}}}_{\text{k-NN}}$	k-NN regressor targeting $\boldsymbol{\mu}^{\text{LF}}$
$\mathcal{M}_{\text{RG}}$	Regression model
$\mathcal{M}^{\mathbf{y}}_{\text{RG}}$	Regression model targeting $\mathbf{y}$
$\mathcal{M}^{\delta}_{\text{RG}}$	Regression model targeting δ
m_A	Axle-carried mass
$\mathbb{N}$	Space of natural numbers
N_{ℓ}	Number of damage classes
N_{C}	Number of chunks

$N_C^{\mathbf{A}}$	Number of columns in $\mathbf{A}$
$N_C^{\mathcal{J}}$	Width of $\mathcal{J}$
N_c^{ℓ}	Size of the cell state of $\mathcal{R}^{\ell}$
$N_{\mathcal{D}}$	Number of possible digital states
N_F^{ℓ}	Number of features channels in the input to C_{ℓ}
N_{FE}	Number of degrees of freedom in the finite element model
N_{folds}	Number of cross-validation folds
N_h	Dimension of the feature space
N_k	Number of nearest neighbors
N_{ℓ}	Number of layers
N_N^{ℓ}	Number of neurons in $\mathcal{F}^{\ell}$
$N_{\mathrm{OC}}^{\mathrm{P}}$	Number of pairs collected in $\mathcal{D}_{\mathrm{OC}}^{\mathrm{P}}$
N_{obs}	Number of observations in the data batch
N_P	Number of samples drawn from the parametric input space
N_{par}	Number of parameters in $\boldsymbol{\mu}$
$N_{\mathrm{par}}^{\mathrm{HF}}$	Number of parameters in $\boldsymbol{\mu}^{\mathrm{HF}}$
$N_{\mathrm{par}}^{\mathrm{LF}}$	Number of parameters in $\boldsymbol{\mu}^{\mathrm{LF}}$
$N_R^{\mathbf{A}}$	Number of rows in $\mathbf{A}$
$N_R^{\mathcal{J}}$	Height of $\mathcal{J}$
N_{RB}	Number of POD basis functions retained in the reduced basis expansion
$N_{\mathrm{RB}}^{\mathbf{A}}$	Number of terms retained in the low-rank approximation of $\mathbf{A}$
N_{S}	Number of snapshots
$N_{\mathcal{S}}$	Number of possible system states
N_{test}	Number of testing instances
$N_{\mathrm{test}}^{\mathrm{HF}}$	Number of HF testing instances
$N_{\mathrm{test}}^{\mathrm{LF}}$	Number of LF testing instances
N_{train}	Number of training instances
$N_{\mathrm{train}}^{\mathrm{HF}}$	Number of high-fidelity training instances
$N_{\mathrm{train}}^{\mathrm{LF}}$	Number of low-fidelity training instances
$N_{\mathrm{train}}^{\mathrm{P}}$	Total number of pairs
$N_{\mathrm{train}}^{\mathrm{RG}}$	Number of instances dedicated to train $\mathrm{NN}_{\mathrm{RG}}^{\delta}$
$N_{\mathcal{U}}$	Number of available actions

N_u	Number of sensor locations
$N_{\mathcal{X}}$	Number of nodes in $\mathcal{X}$
$N_{\boldsymbol{x}}$	Number of observed variables in an energy-based model
$N_{\boldsymbol{y}}$	Number of local factors describing $\mathcal{G}$
$N_{\boldsymbol{y}}$	Number of variables to be inferred in an energy-based model
N_δ	Number of δ intervals
N_θ	Number of parameters in $\boldsymbol{\theta}$
N_ω	Number of connection weights in $\boldsymbol{\Xi}$
N_+	Number of positive pairs
N_-	Number of negative pairs
$N_{\measuredangle}$	Number of quantile bins in the MTF encoding
$\mathrm{NDe}(X_{N_i})$	Non-descendant nodes of X_{N_i}
$\mathrm{NN}_{\mathrm{CAE}}$	Constrastive autoencoder
$\mathrm{NN}_{\mathrm{CL}}$	Classification neural network targeting $\mathbf{b}$
$\mathrm{NN}_{\mathrm{DEC}}$	Decoder branch of an autoencoder
$\mathrm{NN}_{\mathrm{ENC}}$	Encoder branch of an autoencoder
$\mathrm{NN}_{\mathrm{HF}}$	High-fidelity component
$\mathrm{NN}_{\mathrm{HF}}^{\mathrm{only}}$	Single-fidelity counterpart of $\mathrm{NN}_{\mathrm{MF}}$
$\mathrm{NN}_{\mathbf{h}}$	Deep metric learning regression model
$\mathrm{NN}_{\mathrm{LF}}$	Low-fidelity component
$\mathrm{NN}_{\mathrm{MF}}$	Multi-fidelity surrogate model
$\mathrm{NN}_{\mathrm{OC}}$	Deep metric learning (operational conditions) feature extractor model
$\mathrm{NN}_{\mathrm{RG}}^{\mathbf{y}}$	Regression neural network targeting $\mathbf{y}$
$\mathrm{NN}_{\mathrm{RG}}^{\delta}$	Regression neural network targeting δ
$\mathrm{NN}_{\mathrm{SUR}}$	feature-oriented surrogate model
$\boldsymbol{o}^\ell$	Modulating factors in the output gate of $\mathcal{R}^\ell$
o	Observational data
$\mathbf{P}$	Matrix of the left singular vectors of $\mathbf{A}$
$\mathcal{P}_{\max}^\ell$	ℓ-th one-dimensional max pooling layer
$\mathbf{p}_l$	l-th left singular vector of $\mathbf{A}$
p	Probability density function
$\widetilde{p}$	Proposal density

$\widehat{p}$	Unnormalized target posterior distribution
$p_{\max}$	Maximum pressure that a train axle can exert
$p_{\mathbf{H}_\varepsilon}$	Probability density function of $\mathbf{H}_\varepsilon$
$\overline{\mathrm{Pa}}(X_{N_i})$	Parent nodes of X_{N_i}
Q	Load amplitude of $q(t)$
$\widehat{Q}$	Data-driven prediction of Q
q	Quantities of interest
$q(t)$	Distributed harmonic load
$\mathbb{R}$	Space of real numbers
R^2	Coefficient of determination
$\widehat{R}$	Estimated potential scale reduction
R_ε	Tolerance value for $\widehat{R}$
$\mathcal{R}$	Space of rewards
$\mathcal{R}^\ell$	ℓ-th long short-term memory layer
$\mathscr{R}$	Rank of $\mathbf{A}$
$\boldsymbol{\varkappa}^{\mathrm{HF}}$	POD encoding of high-fidelity vibration recordings
$\boldsymbol{\varkappa}^{\mathrm{LF}}$	Weighted POD coefficients encoding low-fidelity vibration recordings
$\widehat{\boldsymbol{\varkappa}}^{\mathrm{LF}}$	Data-driven prediction of $\boldsymbol{\varkappa}^{\mathrm{LF}}$
r	Agent reward
r^{control}	Reward relative to the control inputs
r^{health}	Reward relative to the health state
r^{state}	Reward relative to the system state
$\mathbf{S}$	Snapshot matrix
$\mathbf{S}^{\mathrm{time}}$	Snapshot matrix collecting FOM solutions for a given parameter vector
$\mathbf{S}_{\mathrm{HF}}$	Vectorized high-fidelity data instances in $\mathcal{D}_{\mathrm{HF}}$
$\mathbf{S}_{\mathrm{LF}}$	Vectorized low-fidelity data instances in $\mathcal{D}_{\mathrm{LF}}$
$\mathcal{S}$	Space of system states
s	System state
$\mathbf{T}$	Boolean matrix serving as the connectivity table between the computational model and the sensing system of the structure
T	Finite time horizon
T_q	Linear-ramp actuation time

t	Time variable
t_c	Current time
t_p	Prediction time
$\mathbb{U}$	Utility function
$\overline{\mathbb{U}}$	Expected utility function
$\mathbf{U}$	Vibration recordings instance shaped as multivariate time series
$\widehat{\mathbf{U}}$	Data-driven prediction of $\mathbf{U}$
$\mathbf{U}^{\mathrm{EXP}}$	Query vibration recordings instance
$\mathbf{U}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$	Batch of N_{obs} query vibration recordings
$\mathbf{U}^{\mathrm{HF}}$	High-fidelity vibration recordings instance
$\widehat{\mathbf{U}}^{\mathrm{HF}}$	Data-driven prediction of $\mathbf{U}^{\mathrm{HF}}$
$\mathbf{U}^{\mathrm{LF}}$	Low-fidelity vibration recordings instance
$\widehat{\mathbf{U}}^{\mathrm{LF}}$	Data-driven prediction of $\mathbf{U}^{\mathrm{LF}}$
$\mathcal{U}$	Space of actions
u	Control input
u^A	Actual control input that is effectively executed on the system
u^*	Optimal control input
$\mathbf{V}$	Matrix of nodal displacement values in the time interval $(0, T)$
$\mathbb{V}^{\pi}$	State-value function for policy π
$\mathbb{V}^{\pi*}$	Optimal state-value function
$\mathbf{V}_{\mathrm{HF}}$	High-fidelity nodal displacement values in $(0, T)$
$\mathbf{V}_{\mathrm{LF}}$	Low-fidelity nodal displacement values in $(0, T)$
$\widehat{V}_R$	Estimated between-chain variance of samples
$\mathbf{v}_{\overline{\Xi}}$	Momentum term
$\mathtt{vec}$	Vectorization operation
$\mathbb{W}^{\mathbf{A}}_{N_{\mathrm{RB}}}$	Space of all the possible $N^{\mathbf{A}}_{\mathrm{RB}}$-dimensional orthogonal bases in $\mathbb{R}^{N^{\mathbf{A}}_R}$
$\mathbf{W}$	POD basis matrix embodying time and parametric dependencies of the problem
$\mathbf{W}^{\mathrm{time}}$	POD basis matrix embodying the time dependence of the problem for a single set of input parameters
$\mathbf{W}_{\mathbf{A}}$	POD basis matrix collecting the first $N^{\mathbf{A}}_{\mathrm{RB}}$ columns

	of $\mathbf{P}$
W_R	Average within-chain variance of samples
$\boldsymbol{w}$	Weighting vector underlying $\boldsymbol{r}^{\mathrm{LF}}$
$\mathcal{X}$	Set of nodes
$\mathcal{X}_I$	Set of Interface variables in $\mathcal{G}^{\rightarrow}$
$\mathbf{x}$	Nodal displacement vector
$\widehat{\mathbf{x}}$	Reduced displacement vector
$\mathbf{x}_0$	Nodal displacement initial condition
$\dot{\mathbf{x}}$	Nodal velocity vector
$\dot{\widehat{\mathbf{x}}}$	Reduced velocity vector
$\dot{\mathbf{x}}_0$	Nodal velocity initial condition
$\ddot{\mathbf{x}}$	Nodal acceleration vector
$\ddot{\widehat{\mathbf{x}}}$	Reduced acceleration vector
$\boldsymbol{x}$	Observed variables in an energy-based model
$x^0_{n_A}$	Abscissa of the n_A-th axle at time $t = 0$
x_{n_S}	Abscissa of the axis of the n_S-th sleeper
x_Ω	x coordinate of $\Omega_{\mathbf{y}}$
$\widehat{x}_\Omega$	Data-driven prediction of x_Ω
$\mathbf{Y}_{\mathrm{LF}}$	POD basis matrix collecting the first L_{LF} POD modes of $\mathbf{S}_{\mathrm{LF}}$
$\mathbf{Y}_{\mathrm{OC}}$	POD basis matrix collecting the first L_{OC} POD modes of $\mathbf{S}_{\mathrm{HF}}$
$\mathcal{Y}$	Subset of $\mathcal{X}$
$\mathbf{y}$	Spatial coordinates of the stiffness reduction
$\widehat{\mathbf{y}}$	Data-driven prediction of $\mathbf{y}$
$\boldsymbol{y}$	Variables to be inferred in an energy-based model
y_Ω	y coordinate of $\Omega_{\mathbf{y}}$
$\widehat{y}_\Omega$	Data-driven prediction of y_Ω
$\mathbf{Z}$	Matrix of the right singular vectors of $\mathbf{A}$
$\mathcal{Z}$	Weighted adjacency matrix
$\overline{\mathcal{Z}}$	MTF matrix
$\mathbf{z}_l$	l-th right singular vector of $\mathbf{A}$
z_Ω	z coordinate of $\Omega_{\mathbf{y}}$
$\widehat{z}_\Omega$	Data-driven prediction of z_Ω
$\overline{z}_{i,j}$	Number of bin transitions $\measuredangle_i \rightarrow \measuredangle_j$ between

	consecutive time steps
α	Acceptance ratio
$\boldsymbol{\beta}^{\ell}$	Set of bias terms of $\mathcal{F}^{\ell}$
$\boldsymbol{\beta}^{\ell}_{\square}$	Vectors of learnable bias terms employed in $\mathcal{R}^{\ell}$, for $\square \in \{\psi, \iota, c, o\}$
β^{ℓ}_{n}	Bias term for the n-th neuron of $\mathcal{F}^{\ell}$
Γ	Binary variable to discriminate between positive and negative pairs
γ	Discount factor
δ	Magnitude of the stiffness reduction
$\widehat{\delta}$	Data-driven prediction of δ
$\delta^{\mathrm{K}}_{i\mathcal{J}}$	Kronecker delta
ε	Oriented energy associated with the sequence of vectors forming a matrix
$\varepsilon_{\mathbf{e}}$	Oriented energy in the direction defined by $\mathbf{e}$
$\overline{\epsilon_{\mathbf{h}}}$	Margin distance for the Square-Square loss function
$\epsilon_{\mathbb{V}}$	Tolerance on the utility error
ϵ_{σ}	Tolerance on the fraction of energy content to be disregarded in the POD-based approximation
$\eta_{\mathbf{v}}$	Momentum weight
$\eta_{\overline{\Xi}}$	Learning rate
$\boldsymbol{\Theta}^{\ell}$	Set of learnable kernel filters for C_{ℓ}
$\boldsymbol{\theta}$	Vector of sought parameters
θ	Data label
$\widehat{\theta}$	Data-driven prediction of θ
$\overline{\theta}$	Proposed trial sample of θ
θ_{f}	Entry of $\boldsymbol{\theta}$ encoding f_q
θ_{Q}	Entry of $\boldsymbol{\theta}$ encoding Q
θ_{x}	Entry of $\boldsymbol{\theta}$ encoding x_{Ω}
θ_{z}	Entry of $\boldsymbol{\theta}$ encoding z_{Ω}
θ_{δ}	Entry of $\boldsymbol{\theta}$ encoding δ
θ_{Ω}	Entry of $\boldsymbol{\theta}$ encoding the position of $\Omega_{\mathbf{y}}$
$\boldsymbol{\iota}^{\ell}$	Detection of information update in the input gate of $\mathcal{R}^{\ell}$
Λ	Energy function

$\Lambda_{\boldsymbol{\Omega}}$	Family of energy functions parametrized by $\boldsymbol{\Omega}$
λ_{L^2}	L^2 regularization rate
$\boldsymbol{\mu}$	Vector of system-level parameters
$\boldsymbol{\mu}^{\mathrm{EXP}}$	Query input parameters
$\boldsymbol{\mu}^{\mathrm{HF}}$	High-fidelity input parameters
$\boldsymbol{\mu}^{\mathrm{LF}}$	Low-fidelity input parameters
ν	Poisson's ratio
$\boldsymbol{\Xi}$	Set of connection weights parametrizing a DNN
$\overline{\boldsymbol{\Xi}}$	Set of connection weights and bias terms parametrizing a DNN
$\boldsymbol{\Xi}^{\ell}$	Set of connection weights of $\mathcal{F}^{\ell}$
$\overline{\boldsymbol{\Xi}}^{*}$	Optimal set of connection weights and bias terms parametrizing a DNN
$\boldsymbol{\xi}^{\ell}$	Hidden state of $\mathcal{R}^{\ell}$
π	Control policy
π^{*}	Optimal control policy
ρ	Material density
ρ_B	Ballast layer density
$\boldsymbol{\Sigma}$	Matrix of the singular values of $\mathbf{A}$
$\boldsymbol{\Sigma}_{\varepsilon}$	Covariance matrix of experimental uncertainties
σ_l	l-th singular value of $\mathbf{A}$
σ_{ε}	Standard deviation of experimental uncertainties
ς_{ij}	Correlation coefficient
υ	Train velocity
ϕ	Local factor encoding the joint probability distribution associated with $\mathcal{G}$
ϕ^{control}	Control policy factor
ϕ^{data}	Data assimilation factor
$\phi^{\mathrm{future}\text{-}D}$	State transition probability for future predictions
$\phi^{\mathrm{future}\text{-}R}$	Reward function factor for future predictions
$\phi^{\mathrm{future}\text{-}S}$	State transition probability factor for future predictions
ϕ^{history}	State transition probability factor
ϕ^{NN}	Data assimilation to digital state factor
ϕ^{QoI}	Quantities of interest factor
ϕ^{reward}	Reward function factor

ϕ^{update}	Digital state update factor
χ	Distance weight
χ_R	Weighting factor to tune the decision-maker behavior
$\boldsymbol{\psi}^{\ell}$	Output of the forget gate of $\mathcal{R}^{\ell}$
$\boldsymbol{\Omega}_{\text{AE}}$	Learnable weights parametrizing NN_{ENC} and NN_{DEC}
$\boldsymbol{\Omega}_{\text{CAE}}$	Learnable weights parametrizing NN_{CAE}
$\boldsymbol{\Omega}_{\text{CL}}$	Learnable weights parametrizing NN_{CL}
$\boldsymbol{\Omega}_{\text{HF}}$	Learnable weights parametrizing NN_{HF}
$\boldsymbol{\Omega}_{\mathbf{h}}$	Learnable weights parametrizing $\text{NN}_{\mathbf{h}}$
$\boldsymbol{\Omega}^{\ell}_{h\square}$	Matrices of learnable weights applied to the input features of $\mathcal{R}^{\ell}$, for $\square \in \{\psi, \iota, c, o\}$
$\boldsymbol{\Omega}_{\text{LF}}$	Learnable weights parametrizing NN_{LF}
$\boldsymbol{\Omega}_{\text{OC}}$	Learnable weights parametrizing NN_{OC}
$\boldsymbol{\Omega}_{\text{RG}}$	Learnable weights parametrizing NN_{RG}
$\boldsymbol{\Omega}_{\text{SUR}}$	Learnable weights parametrizing NN_{SUR}
$\boldsymbol{\Omega}^{\ell}_{\xi\square}$	Matrices of learnable weights applied to the hidden state of $\mathcal{R}^{\ell}$, for $\square \in \{\psi, \iota, c, o\}$
$\Omega_{\mathscr{b}}$	Damage subdomain associated with $\mathscr{b}$
$\Omega_{\mathbf{y}}$	Damage subdomain
$\boldsymbol{\omega}^{\ell}_{n}$	Connection weights for the n-th neuron of $\mathcal{F}^{\ell}$
$\curlywedge$	MTF quantile bins
$\Phi^{\ell}_{\mathcal{P}}$	Stride width of $\mathcal{P}^{\ell}_{\max}$
$\odot$	Hadamard element-wise product
$\perp$	Conditional independence
$\circ$	Function composition
$\lVert\cdot\rVert_1$	1-norm
$\lVert\cdot\rVert_2$	2-norm
$\lVert\cdot\rVert_F$	Frobenius norm

ACRONYMS

ACC: classification accuracy
AE: autoencoder
API: application programming interface
BN: batch normalization
CAE: contrastive autoencoder
CNN: convolutional neural network
CPT: conditional probability table
CPU: central processing unit
DBN: dynamic Bayesian network
DDN: dynamic decision network
DL: deep learning
DN: do nothing
DNN: deep neural network
DOFs: degrees of freedom
DT: digital twin
Dpout: Dropout
EOVs: environmental and operational variations
EPSR: estimated potential scale reduction
FE: finite element
FN: false negatives
FOM: full-order model
FP: false positives
GPU: graphics processing unit
HF: high-fidelity
LF: low-fidelity
LSTM: long short-term memory
MA: major imperfect maintenance
MAE: mean absolute error
MCMC: Markov chain Monte Carlo

MDP: Markov decision process
MDS: multidimensional scaling
MF: multi-fidelity
MI: minor imperfect maintenance
ML: machine learning
MSE: mean squared error
MTF: Markov transition field
NN: neural network
OC: operational conditions
PC: personal computer
PCC: Pearson correlation coefficient
PDF: probability density function
PGM: probabilistic graphical model
PM: perfect maintenance
POD: proper orthogonal decomposition
POM: proper orthogonal mode
PReLU: parametric rectified linear unit
RAM: random access memory
RE: restrict operational conditions
RMS: root mean square
RMSE: root mean squared error
RNN: recurrent neural network
ROM: reduced-order model
ReLU: rectified linear unit
SHM: structural health monitoring
SM: surrogate model
SNR: signal-to-noise ratio
SVD: singular value decomposition
StdDev: Standard deviation
TN: true negatives
TP: true positives
Tanh: hyperbolic tangent
k-NN: k-nearest neighbors

LIST OF FIGURES

1.1 Digital model, digital shadow, and digital twin 3
1.2 Vibration-based SHM: simplified scheme 7
1.7 Graphical outline of the Thesis 29

4.1 Example fully-connected layer 67
4.2 Example convolutional neural network architecture 78
4.3 LSTM cell 79
4.4 Example autoencoder architecture 81
4.5 Example Bayesian networks 86
4.6 Example dynamic Bayesian network 89
4.7 Agent-environment interaction 91
4.8 Example decision networks 95

5.1 Deep metric learning: graphical abstraction 104
5.2 L-shaped cantilever beam: monitoring setup 109
5.3 L-shaped beam: $\mathrm{NN}_{\mathbf{h}}$ embeddings for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 111
5.4 L-shaped beam: prediction errors for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 112
5.5 L-shaped beam: multidimensional scaling for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 114
5.6 L-shaped beam: training embeddings for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 115
5.7 L-shaped beam: testing embeddings for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 116
5.8 L-shaped beam: prediction errors for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 117
5.9 L-shaped beam: multidimensional scaling for $\mathcal{L}_{\mathbf{h}}^{(3)}$ 118
5.10 L-shaped beam: normalized embeddings for $\mathcal{L}_{\mathbf{h}}^{(3)}$ 119
5.11 L-shaped beam: prediction errors for $\mathcal{L}_{\mathbf{h}}^{(3)}$ 120
5.12 Hörnefors railway bridge 122
5.13 Railway bridge: geometrical data 123
5.14 Gröna Tåget train type 124
5.15 Railway bridge: monitoring setup 124
5.16 Railway bridge: finite element discretization 125
5.17 Railway bridge: time-modulating function. 126
5.18 Railway bridge: example singular values spectrum 127
5.19 Railway bridge: FOM and ROM comparison 127
5.20 Railway bridge: $\mathrm{NN}_{\mathbf{h}}$ embeddings for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 128
5.21 Railway bridge: prediction errors for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 130
5.22 Railway bridge: multidimensional scaling for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 131

5.23 Railway bridge: prediction errors for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 132

6.1 Multi-fidelity surrogate: graphical abstraction 137
6.2 Multi-fidelity surrogate: low-fidelity component 139
6.3 Multi-fidelity surrogate: high-fidelity component 140
6.4 L-shaped beam: example multi-fidelity results 145
6.5 L-shaped beam: example single-fidelity results 146
6.6 L-shaped beam: $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ learning curves 148
6.7 Railway bridge: monitoring setup 149
6.8 Railway bridge: example multi-fidelity results 150
6.9 Railway bridge: example single-fidelity results 151
6.10 Railway bridge: $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ learning curves 152
6.11 Multi-fidelity surrogate: online SHM phase 153
6.12 Siamese architecture 154
6.13 L-shaped beam: $\mathrm{NN}_{\mathrm{OC}}$ learning curve 155
6.14 L-shaped beam: preliminary identification 157
6.15 Multi-fidelity surrogate: MCMC procedure 159
6.16 L-shaped beam: example MCMC outcome 160

7.1 Enhanced Bayesian update: graphical abstraction 166
7.2 Feature extractor and feature-oriented surrogate 170
7.3 Enhanced Bayesian Model Updating 173
7.4 L-shaped beam: example MTF mosaic 175
7.5 L-shaped beam: 3D multidimensional scaling 176
7.6 L-shaped beam: example MCMC outcome 177
7.7 Portal frame: monitoring setup 178
7.8 Portal frame: 3D multidimensional scaling 179
7.9 Portal frame: example MCMC outcome 181
7.10 Railway bridge: 3D multidimensional scaling 182
7.11 Railway bridge: example MCMC outcome 184

8.1 Predictive digital twin: graphical abstraction 186
8.2 Predictive digital twin: dynamic decision network 188
8.3 Predictive digital twin: prediction graph 192
8.4 L-shaped cantilever beam: monitoring setup 197
8.5 L-shaped beam: confusion matrix 198
8.6 L-shaped beam: control policy with 2 actions 201
8.7 L-shaped beam: online phase with 2 actions 202
8.8 L-shaped beam: future prediction with 2 actions 202
8.9 L-shaped beam: control policy with 4 actions 204
8.10 L-shaped beam: online phase with 4 actions 205
8.11 L-shaped beam: future prediction with 4 actions 205
8.12 Railway bridge: monitoring setup 207
8.13 Railway bridge: confusion matrix 208
8.14 Railway bridge: control policy 209
8.15 Railway bridge: online phase 210

LIST OF TABLES

5.1 L-shaped beam: damage location accuracy for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 113
5.2 L-shaped beam: damage location accuracy for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 114
5.3 L-shaped beam: damage location accuracy for $\mathcal{L}_{\mathbf{h}}^{(3)}$ 120
5.4 L-shaped beam: robustness assessment 121
5.5 Railway bridge: damage location accuracy for $\mathcal{L}_{\mathbf{h}}^{(1)}$ 129
5.6 Railway bridge: damage location accuracy for $\mathcal{L}_{\mathbf{h}}^{(2)}$ 130

6.1 L-shaped beam: $\mathrm{NN}_{\mathrm{LF}}$ performance 144
6.2 L-shaped beam: $\mathrm{NN}_{\mathrm{HF}}$ performance 147
6.3 Railway bridge: $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ performance 149
6.4 L-shaped beam: preliminary identification 156
6.5 L-shaped beam: damage localization outcomes 159
6.6 L-shaped beam: preliminary identification impact 161

7.1 L-shaped beam: damage localization outcomes 177
7.2 Portal frame: damage identification outcomes 180
7.3 Railway bridge: damage identification outcomes 183

A.1 $\mathrm{NN}_{\mathbf{h}}$ architecture and hyperparameters 263
B.1 $\mathrm{NN}_{\mathrm{LF}}$ architecture and hyperparameters 265
B.2 $\mathrm{NN}_{\mathrm{HF}}$ architecture and hyperparameters 266
B.3 $\mathrm{NN}_{\mathrm{OC}}$ architecture and hyperparameters 266
D.1 $\mathrm{NN}_{\mathrm{CAE}}$ architecture and hyperparameters 271
D.2 $\mathrm{NN}_{\mathrm{SUR}}$ architecture and hyperparameters 272
E.1 $\mathrm{NN}_{\mathrm{CL}}$ architecture and hyperparameters 274
E.2 $\mathrm{NN}_{\mathrm{RG}}$ architecture and hyperparameters 274

This page intentionally left blank.

1 INTRODUCTION

1.1 WHY PREDICTIVE DIGITAL TWINS OF STRUCTURES?

The optimal management of deteriorating structural systems is an increasing challenge in modern engineering. While the demand for functionality, robustness, resilience, durability, and safety of engineering systems continues to grow, their failure or non-optimized maintenance planning may entail high safety, economic, and social costs. Virtual capabilities that can simulate these systems with adequate levels of fidelity, speed, and granularity hold the promise to decrease these costs. The underlying interest in understanding and predicting the behavior of engineering systems has thus motivated the rise of *computational models* for numerically solving problems in engineering.

Numerical simulations provide a firm framework to tackle several tasks, including discovery, optimization, parameter estimation, and decision support. In structural mechanics, numerical methods are employed to approximate the behavior of structures, characterized by geometrical, material, and boundary properties too complex for regular analytical solutions. This capability has lead to innovative approaches in structural design by simulating new configurations to meet regulatory safety requirements for structural components and systems under varying operational conditions, as well as to assess fatigue and damage tolerances, among others.

Computational models alone can only provide insights depending on the model accurately reflecting the underlying system. Uncertainties arising from modeling choices, such as material properties, and environmental and operational variabilities, are among the many factors that can hamper the model capability to faithfully describe a real-world system. For the same reason, and due to their intrinsic evolving nature resulting from aging, material deterioration, fatigue, inappropriate usage, varying operational life, and exposure to loads not accounted for at design stage*, it is also unlikely that the behavior of two real-world systems will overlap, even if intended to do so.

*Extreme loading conditions are increasingly triggered by environmental risks and natural hazards associated with climate change, including anomalous heat waves, droughts, floods, forest fires and destructive windstorms.

Computational models that ignore the uncertain description and the varying nature of engineering systems, both evolving throughout the system life cycle, crucially limit the level of

personalization and, thus, the accuracy of the model and any insights gained through its use. To this end, the *digital twin* (DT) concept has emerged in the last decade as a paradigm to enable diagnostic and predictive capabilities not achievable with computational models alone. The following definition of DT is adopted throughout this Thesis† [1,2,3,4,5,6,7,8,9,10]:

> "A digital twin is a personalized virtual representation of a unique physical asset that evolves over time. It relies on a set of computational models that are dynamically updated to persistently mirror the physical counterpart throughout its life cycle. Additionally, it has predictive capabilities that inform actions tailored to realize value."

†There is currently no consensus within the engineering community about what defines a digital twin. The proposed definition attempts to unify some of the most widely adopted interpretations, stressing what distinguishes a digital twin from a digital model of an asset.

1. Digital Twin: Definition & Value — An AIAA and AIA Position Paper, *AIAA Digital Engineering Integration Committee and others*. 2020

2. Tuegel *et al.*, *Reengineering Aircraft Structural Life Prediction Using a Digital Twin*. 2011

3. Tuegel, *The Airframe Digital Twin: Some Challenges to Realization*. 2012

4. Niederer *et al.*, *Scaling digital twins from the artisanal to the industrial*. 2021

5. Moya *et al.*, *Physically sound, self-learning digital twins for sloshing fluids*. 2020

6. Thelen *et al.*, *A comprehensive review of digital twin – part 1: modeling and twinning enabling technologies*. 2022

7. Grieves *et al.*, *Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems*. 2017

8. Chinesta *et al.*, *Virtual, Digital and Hybrid Twins: A New Paradigm in Data-Based Engineering and Engineered Data*. 2020

9. Rasheed *et al.*, *Digital twin: Values, challenges and enablers from a modeling perspective*. 2020

10. National Academies of Sciences, Engineering, and Medicine, *Foundational Research Gaps and Future Directions for Digital Twins*. 2023

By combining computational models with data-driven learning, the DT paradigm aims to overcome the aforementioned limitations in monitoring, analyzing, and optimizing complex systems. In particular, several crucial aspects characterizing a successful DT can be highlighted:

- The DT paradigm aims for an adaptive and comprehensive virtualization of a unique physical asset throughout its life cycle. Emphasizing the concepts of personalization and adaptivity is crucial here. The conceptualization of a DT should start with an initial calibration to the as-manufactured asset. The DT should then extend beyond calibration into the service life of the asset once it enters operation. As the structural state begins to evolve over time, e.g., due to damage, gradual degradation, or maintenance events, the unique characteristics of a physical asset vary at specific points in its life cycle. Capturing this variability among assets and across the asset life cycle is the goal, achieved by creating, calibrating, and dynamically evolving a DT for each unique asset.

- The computational models and parameters included in a DT are continually updated through the assimilation of real-world data, or sensor recordings. Models play a crucial role in ensuring accurate representation of the asset and serve as a platform for analysis, simulation, and optimization. These models take the form of either data-driven models, or physics-based simulators, or a combination of the two. As such, both computational models and data are critical enablers for successful DTs. This perspective requires a synergistic coupling between the physical asset and its virtualization, resulting in an asset-twin coupled dynamical system. This connection is illustrated in Fig. 1.1, where the fully-automated bi-directional information flow characterizing a DT‡ is juxtaposed with the *digital model*

‡The end-to-end connection characterizing a digital twin involves a from-physical-to-digital information flow, through data assimilation and inference, and a from-physical-to-digital information flow, in the form of informed control actions.

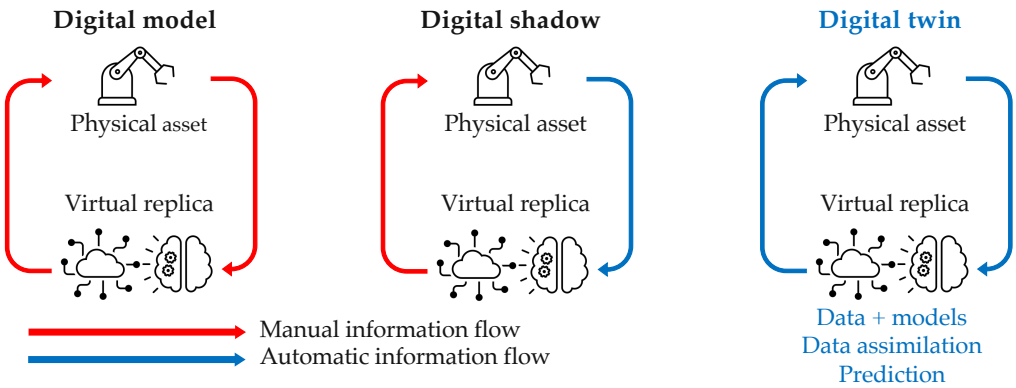


Figure 1.1: Digital model, digital shadow, and digital twin: manual and automatic information flows.

and the *digital shadow* cases. While the digital model lacks any interaction with the physical asset, the digital shadow enjoys data assimilation capabilities, allowing it to stay synchronized with its physical counterpart by means of data-collecting devices. However, both the digital model and the digital shadow lack an automated interaction mechanism targeting the physical asset, thus preventing the realization of added value. This latter aspect, enabled by an automatic controller or with human in the loop, is what propels the DT concept toward autonomy instead of simple automation.

- While many decisions are made throughout the life cycle of a physical asset, they traditionally rely on experiential insights into the variables influencing the performance level. However, up-to-date and comprehensive models are suitable to be exploited within *dynamic decision-making* frameworks. Observational data can be employed for descriptive, diagnostic, and predictive analytics to automate informed decision-making tailored to the specific asset at every life cycle phase. To this aim, the monitoring of structural health and the prediction of future conditions of the asset can enable decisions regarding its maintenance, repair, inspection, and management.

- Enhanced computational efficiency is essential to handle the continuous assimilation of noisy and big data, as well as to accommodate uncertainty quantification and propagation. The uncertainty in the estimated state or in the external variables affecting the physical asset has to be propagated into both present and future performance predictions and decision-making processes.

Originating from the aeronautical and aerospace engineering community[§], the DT concept has emerged in various scientific

[§]Many consider the Apollo program a pioneer in implementing the digital twin concept, through the use of on-ground simulators to replicate the systems deployed in space. During the Apollo 13 mission, which experienced an in-flight explosion causing critical damage to the spacecraft engine, NASA mission controllers used data from the damaged spacecraft to update the simulators, enabling them to make informed decisions that ultimately ensured the safe return of the astronauts.

areas for operational monitoring, control, and decision support. DT applications span structural health monitoring (SHM) and predictive maintenance [11,12,13], additive manufacturing [14], smart cities [15], urban sustainability [16], railway systems management [17], and unmanned aerial vehicles [18,19,20]. Beyond engineering, the DT paradigm is also being deployed in healthcare for advancing medical diagnosis and personalized treatment [21,22,23], climate science [24], and augmented reality [25], among others. These instances showcase the potential of DTs to serve as the foundation for intelligent automation in the future, facilitating asset-specific analysis and decision-making.

The above discussion emphasizes that while DTs inherently demand application-specific requirements, there are fundamental principles common to the DT concept itself. Another noteworthy aspect is the multidisciplinary nature characterizing the DT perspective, encompassing processes such as data assimilation, parameter estimation, and inference, all targeted at DT updating. This journey continues with the use of computational models to gain insights into both present and future asset performance, concluding with the optimization of state-dependent control policies to steer the state of the asset in a way that maximizes the expected future performance.

Within an SHM framework, adopting a twinning perspective can be achieved through the assimilation of sensor data using data-driven structural health diagnostics (*from-physical-to-digital* information flow), possibly accommodating the quantification and propagation of aleatory and epistemic uncertainties. The up-to-date digital replica of the structural system should then enable the prediction of its evolution and inform optimal planning for maintenance and management actions (*from-digital-to-physical* information flow). Enabling a DT perspective for a critical structural system, whether for safety or operative reasons, can yield numerous benefits throughout its entire life cycle and is crucial to allow for *condition-based* or *predictive* maintenance practices, in place of customarily employed *time-based* ones.

This Thesis aims to advance DT technologies within the framework of SHM and predictive maintenance by leveraging on the extreme modeling capabilities enabled by deep learning (DL) models. In the following Sections, we further elaborate on the framework through which this Thesis aims at providing a fresh contribution. We delve into the need for SHM in Sec. 1.2, and the need for physics-based reduced-order and data-driven surrogate modeling in Sec. 1.3. Throughout this introductory Chapter, we identify the challenges and opportunities, from which we finally formulate the objectives of this Thesis in Sec. 1.4.

11. Li *et al.*, *Dynamic Bayesian Network for Aircraft Wing Health Monitoring Digital Twin*. 2017

12. Ye *et al.*, *Digital twin for the structural health management of reusable spacecraft: a case study*. 2020

13. Podskarbi *et al.*, *Digital Twin for Operations - Present Applications and Future Digital Thread*. 2020

14. Phua *et al.*, *A digital twin hierarchy for metal additive manufacturing*. 2022

15. Jans-Singh *et al.*, *Digital twin of an urban-integrated hydroponic farm*. 2020

16. Tzachor *et al.*, *Potential and limitations of digital twins to achieve the Sustainable Development Goals*. 2022

17. Arcieri *et al.*, *Bridging POMDPs and Bayesian decision making for robust maintenance planning under model uncertainty: An application to railway systems*. 2023

18. McClellan *et al.*, *A Physics-Based Digital Twin for Model Predictive Control of Autonomous Unmanned Aerial Vehicle Landing*. 2022

19. Kapteyn *et al.*, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*. 2021

20. Sisson *et al.*, *Digital Twin Approach for Component Health-Informed Rotorcraft Flight Parameter Optimization*. 2022

21. Corral-Acero *et al.*, *The 'Digital Twin' to enable the vision of precision cardiology*. 2020

22. Chaudhuri *et al.*, *Predictive digital twin for optimizing patient-specific radiotherapy regimens under uncertainty in high-grade gliomas*. 2023

23. Bruynseels *et al.*, *Digital Twins in Health Care: Ethical Implications of an Emerging Engineering Paradigm*. 2018

24. Bauer *et al.*, *A digital twin of Earth for the green transition*. 2021

25. Moya *et al.*, *Digital twins that learn and correct themselves*. 2022

Challenges in the life cycle of structural systems include aging, deterioration processes due to chemical attacks and physical damage mechanisms, inappropriate usage, unexpected loading conditions, and natural hazards, among others. Coupled with the possibility of poor or limited maintenance, the detrimental effects of these phenomena can lead over time to unsatisfactory structural performance even under service loadings [26]. Exacerbated by the impacts of climate change [27], experts estimate that annual damages to critical infrastructures in energy, transport, industrial, and social sectors in Europe could increase from the current 3.4 to 34 billion € by 2100 [28]. The costs associated with structural failures are not solely related to users safety and rehabilitation. Failures of bridges and buildings also have a significant impact on public opinion, raising concerns for policymakers and regulators. Traffic delays and lost productivity due to service interruption of infrastructures hamper the economic growth and cost users billions of dollars annually [29]. For instance, a potential shutdown of the Hudson River rail tunnel in New York City was estimated to cost the United States economy about $16 billion in 2019 [30], with more than half attributed to the time lost by users during longer daily commutes.

In 2001, The European Union project "Bridge Management in Europe" identified that highway bridges in France, Germany and the United Kingdom present deficiencies at a rate of 39%, 30% and 37%, respectively [31]. In 2021, the United States had over 600,000 bridges. Among these, 42% were at least 50 years old, and more than 46,000 of them were deemed structurally deficient, meaning they were in a "poor" condition according to [32] — bridges that require significant maintenance, rehabilitation, or replacement. Unfortunately, these structurally deficient bridges are crossed about 180 million times every day. This situation underscores the need for a systematic bridge preservation program, prioritizing existing deterioration and emphasizing preventive maintenance. The reason is twofold. First, the positive trend in repair investments indicates a shift in emphasis from building new bridges to maintaining existing ones. Second, there is a need of preserving bridges in fair condition, as addressing issues at this stage incurs only a fraction of the cost required for structurally deficient bridges.

In this context, adopting a DT perspective is crucial to enable the transition from customarily employed time-based maintenance practices relying on scheduled in situ inspections* to *condition-based* and *predictive* maintenance paradigms [33,34].

26. Biondini *et al.*, *Life-Cycle Performance of Deteriorating Structural Systems under Uncertainty: Review*. 2016

27. Civil Engineers, *Adaptive Design and Risk Management*. 2018

28. Mariniello *et al.*, *Escalating impacts of climate extremes on critical infrastructures in Europe*. 2018

29. Thoft-Christensen, *Life-cycle cost-benefit (LCCB) analysis of bridges from a user and social point of view*. 2009

30. Jones *et al.*, *A preventable crisis: The economic and human costs of a Hudson River rail tunnel shutdown*. 2019

31. Gkoumas *et al.*, *Research and innovation in bridge maintenance, inspection and monitoring*. 2019

32. ASCE, *A comprehensive assessment of America's infrastructure*. 2021

*Scheduled in situ inspections involve visual checks, possibly enhanced by non-destructive evaluation techniques, to assess the local properties of a material, component, or system without causing damage. However, in situ inspections alone can not fully address the need for safe structures and infrastructures. This not only due to the potential limitations in reliability and accuracy resulting from their localized nature and available expertise, but also because the assessed structural performance pertains to the specific moment of the inspection.

33. Glaser *et al.*, *Sense of Sensing: From Data to Informed Decisions for the Built Environment*. 2008

34. Achenbach, *Structural health monitoring – What is the prescription?* 2009

Condition-based philosophies involve monitoring the system response using data-collecting devices, enabling corrective actions to be taken when damage evolves to a critical level. On the other hand, *predictive* monitoring or maintenance practices involve using current estimations of structural health to predict when the damage will reach a critical level.

This paradigm shift holds the potential to enable personalized monitoring, management and maintenance programs, resulting in a more informed assessment of structural safety, better allocation of available resources, and increased system availability. These advantages in the life cycle are often challenging to quantify. However, the utility of SHM can be quantified using the concept of value of information [35,36,37], which provides a rational metric to assess the benefits associated with gathering information from an SHM system. The associated utility stems from reducing the uncertainty related to the knowledge about the structural state, supporting informed decision-making.

35. Kamariotis *et al.*, *Value of information from vibration-based structural health monitoring extracted via Bayesian model updating*. 2022

36. Andriotis *et al.*, *Value of structural health information in partially observable stochastic environments*. 2021

37. Malings *et al.*, *Value-of-information in spatio-temporal systems: Sensor placement and scheduling*. 2018

Embracing the development of relatively inexpensive hardware components, as well as the Internet of Things, this transition is driven by the synergistic use of pervasive sensor systems and systematic diagnostic and prognostic activities. In this context, SHM has become a standard and widespread framework for damage identification and structural health tracking. Vibration-based SHM techniques in particular, which rely on vibration response data acquired through deployed sensing systems, are well suited for implementing continuous and global monitoring in place of less practical in situ inspections.

Before delving into an analysis of available vibration-based SHM approaches to damage assessment, a fundamental question arises: *What is damage?* Damage arises from the evolution of defects that are naturally present at material level, such as voids, inclusions, dislocations, and micro-cracks, depending on the case. Under appropriate loading scenarios, this material-level damage can propagate, initially affecting the individual components and eventually the entire system. Failure occurs when the damage progresses to a point where the system can no longer perform its intended function. This process can occur on different time scales depending on the cases. Damage can accumulate incrementally over long periods, for instance in the case of fatigue or corrosion, or progress very rapidly, as in the case of unstable propagation of a critical fracture or due to discrete events such as earthquakes and floods. Our interest is restricted to *damage patterns* characterized by a small evolution rate, whose prompt identification can enable a more informed assessment of structural safety.

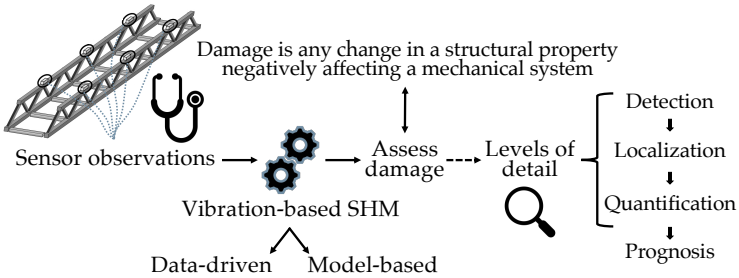


Figure 1.2: Vibration-based structural health monitoring: simplified scheme and hierarchical assessment of damage.

Defining damage as a material internal variable at the microstructural level is impractical when it comes to identifying damage in large-scale mechanical systems. In most practical scenarios, damage manifests itself as a change in stiffness, mass, energy dissipation characteristics, or boundary conditions of the system, leading to alterations in the structure's dynamic response. Throughout this Thesis, we adopt the definition of damage provided by Farrar and Worden [38] in the context of damage identification:

38. Farrar *et al.*, *Structural Health Monitoring: A Machine Learning Perspective*. 2013

> "Damage is defined as intentional or unintentional changes to the material and/or geometric properties, including changes to the boundary conditions and system connectivity, which adversely affect the current or future performance of the system."

A simplified scheme of the SHM process is illustrated in Fig. 1.2. The structure is continuously monitored by assimilating multivariate time series describing vibration recordings, e.g., in terms of displacements or accelerations. According to Rytter [39], the objective of damage identification can be hierarchical, encompassing four levels of increasing detail and assessment complexity: damage *detection*, *localization*, *quantification*, and *prognosis* (assessment of the *remaining useful life*). Among vibration-based techniques, two main approaches are distinguished: *model-based* and *data-driven* [38]. Given the challenge of obtaining knowledge on the structural response of civil structure in presence of varying damage conditions, both approaches typically rely on physics-based simulations, either in the form of models or data.

39. Rytter, *Vibrational Based Inspection of Civil Engineering Structures*. 1993

1.2.1 Simulation-based paradigm

The conceptual underpinning of the *simulation-based paradigm* can be grasped by examining the resolution of inverse problems. In this context, the aim is to estimate the causal factors behind a set of incoming observations by comparing them with available knowledge. Consider, for instance, visual inspections and non-destructive evaluations, where observational data are used to formulate a diagnosis of ongoing damage mechanisms. At

this level, a limited number of damage mechanisms is typically recognizable. The diagnosis relies on comparing observational data with the expected patterns in known cases of damage and understanding their impact on the local properties that influence the mechanical behavior of the system. Applying this intuition to vibration-based SHM, it is evident how impractical and ineffective it would be to have an expert continuously monitoring the data inflow. Automating the processes of extracting *damage-sensitive information* from raw data and using this information for the purpose of damage assessment is therefore crucial. However, a significant challenge arises due to the limited or absent experimental knowledge regarding the effects of damage on structural responses. Indeed, it is almost impossible to conduct systematic destructive experiments on civil structures to accumulate knowledge involving various damage states.

Physics-based simulations usually rely on discretized partial differential equations modeling the structural behavior. These models are typically solved using computational approaches like the finite element (FE) method. The reliability and predictive capability of these methods have been established through decades of research [40,41]. The simulation-based paradigm of SHM [42] provides a knowledge platform where the effect of damage on the structural response can be systematically reproduced using physics-based models. This simulation-based knowledge can take the form of either the model itself, or a database that collects data pertaining to specific damage conditions along with the parameters underlying their generation.

Referring to civil structures, our focus is on damage patterns characterized by a small evolution rate. In this context, a *local reduction of the effective stiffness* stands as the simplest damage description, resulting from a time scale separation between damage evolution and damage assessment. Modeling the changes in the structural dynamic response due to damage by reducing the stiffness properties is a standard procedure in the literature of civil SHM. Indeed, the duration of each observation window is typically small compared to the time necessary for the advancement of the damage process*, allowing the structural behavior to be treated as linear within the observation interval [43].

This approach does not limit the types of damage patterns that can be represented. Examples include the failure of bolted connections and corrosion in steel structures [44,45], cracking and delamination in composite structures [46], cracking in reinforced concrete structures [45,47], and the formation of plastic hinges and cutting of tendons in post-tensioned reinforced concrete structures [48]. Other instances involve cracking and

40. Oden *et al.*, *Predictive Computational Science: Computer Predictions in the Presence of Uncertainty*. 2017

41. Babuška *et al.*, *The Finite Element Method and Its Reliability*. 2001

42. Taddei *et al.*, *Simulation-Based Classification; a Model-Order-Reduction Approach for Structural Health Monitoring*. 2018

*In steel structures, the main causes of long-term material degradation are corrosion, which leads to a gradual reduction in cross-section, and fatigue. In concrete structures, the main causes of lifetime deterioration include chemical processes associated with carbonation, sulphate and chloride attacks, reinforcing steel corrosion, and alkali-silica reactions, as well as physical processes due to freeze/thaw cycles, thermal cycles, and mechanical processes like cracking, abrasion, and fatigue.

43. Eftekhar Azam *et al.*, *Online damage detection in structural systems via dynamic inverse analysis: A recursive Bayesian approach*. 2018

44. Pandey *et al.*, *Damage Detection in Structures Using Changes in Flexibility*. 1994

45. Mariniello *et al.*, *Structural damage detection and localization using decision tree ensemble and vibration data*. 2021

46. Kapteyn *et al.*, *Data-driven physics-based digital twins via a library of component-based reduced-order models*. 2020

45. Mariniello *et al.*, *Structural damage detection and localization using decision tree ensemble and vibration data*. 2021

47. Teughels *et al.*, *Damage assessment by FE model updating using damage functions*. 2002

48. Giglioni *et al.*, *A domain adaptation approach to damage classification with an application to bridge monitoring*. 2024

crushing-induced losses of stiffness in masonry brick structures [49], cuts in steel beams [50], and the erosion of piles and abrasion of deck surfaces [51]. Similarly, changes in boundary conditions, such as modified abutments interactions and pier settlements, can be described by reducing the stiffness of boundary spring elements [51]. Characterizing the specific type of damage that has emerged can be addressed subsequent to the early identification through a detailed in situ inspection.

While the loss of effective stiffness enables representing various damage patterns, there are manifestations of damage that can not be described in this way. Loss of strength, fatigue, diminished ductility, and softening are all damage scenarios that can not be adequately captured through linear structural models [52,53]. For instance, the bond slip of reinforcing bars requires a meticulous modeling approach due to its impact on the ultimate strength of the system, despite a limited change in stiffness. Similarly, representing damage as a local stiffness reduction may underestimate the impact of rusting on the strength reduction in reinforced concrete members. Furthermore, the accumulation of plastic strains in steel and reinforced concrete requires the use of plastic constitutive models [54]. For example, the nonlinear analysis of corroded reinforced concrete structures can refer to lumped or distributed plasticity beam FE models [55].

1.2.2 Model-based and data-driven approaches

Among vibration-based SHM techniques, two main approaches can be highlighted: the *model-based approach* and the *data-driven approach* [38]. In both cases, the inverse problem associated with the SHM task is addressed by comparing incoming observational data with the existing knowledge for known structural conditions. The difference between the two approaches lies in the way existing knowledge is encoded and exploited. Model-based approaches rely on forward models, while data-driven approaches utilize such knowledge in the form of data.

Model-based approaches assess damage by exploiting *physics-based forward models*. In this case, the SHM task is addressed by calibrating the forward model, often based on FE, through filtering [56,57,58,59], sampling [60,61,62,63,64], or optimization [65,66,67,68] techniques. Besides allowing for interpretable results, model-based approaches enable a mechanical intuition of the structural degradation process and forecasting of the system evolution (prognosis). The use of physics-based forward models also makes model-based approaches particularly valuable when the goal of damage assessment extends beyond simple damage detection to damage localization and quantification.

49. Hernández-González *et al.*, *Towards a comprehensive damage identification of structures through populations of competing models*. 2024

50. Seventekidis *et al.*, *Structural Health Monitoring using deep learning with optimal finite element model generated data*. 2020

51. Fernandez-Navamuel *et al.*, *Supervised Deep Learning with Finite Element simulations for damage identification in bridges*. 2022

52. Ebrahimian *et al.*, *Nonlinear finite element model updating for damage identification of civil structures using batch Bayesian estimation*. 2017

53. Giagopoulos *et al.*, *Structural health monitoring and fatigue damage estimation using vibration measurements and finite element model updating*. 2019

54. Luca Rosfalco, *Blending physics and data in structural health monitoring*. 2022

55. Biondini *et al.*, *Deteriorating beam finite element for nonlinear analysis of concrete structures under corrosion*. 2015

38. Farrar *et al.*, *Structural Health Monitoring: A Machine Learning Perspective*. 2013

56. Cristiani *et al.*, *Fatigue damage diagnosis and prognosis of an aeronautical structure based on surrogate modelling and particle filter*. 2021

57. Eftekhar Azam *et al.*, *Online damage detection via a synergy of proper orthogonal decomposition and recursive Bayesian filters*. 2017

58. Corigliano *et al.*, *Parameter identification in explicit structural dynamics: Performance of the extended Kalman filter*. 2004

59. Rosafalco *et al.*, *System Identification via Unscented Kalman Filtering and Model Class Selection*. 2024

60. Hou *et al.*, *Vibration-Based Structural Damage Detection Using Sparse Bayesian Learning Techniques*. 2022

61. Green *et al.*, *Bayesian and Markov chain Monte Carlo methods for identifying nonlinear systems in the presence of uncertainty*. 2015

62. Ierimonti *et al.*, *Bayesian-Based Damage Assessment of Historical Structures Using Vibration Monitoring Data*. 2021

The calibration is carried out by minimizing an error metric between measured and simulated responses. In this respect, *operational modal analysis* is often employed to extract structural modal properties including resonant frequencies, mode shapes, and damping ratios, from periodically acquired ambient vibration data. These properties are then used in place of large amounts of raw noisy data that can potentially compromise the effectiveness of model-based approaches. Moreover, there is no need to know or measure operational conditions, which are often unknown, as in the case of ambient vibrations due to low-energy seismicity of natural or anthropic sources.

However, the associated inverse problem is not only ill-posed* [69], but also encompasses a full range of uncertainties related to, e.g., measurement noise, modeling assumptions, and environmental or operational variabilities. Therefore, a probabilistic framework enabled by *Bayesian model updating*, in the form of either filtering or sampling algorithms, appears to be one of the most natural ways to address the foreseen parameter estimation problem. Bayesian strategies update the *prior belief* about SHM parameters to the corresponding *posterior* probability density function (PDF), quantifying the relevant uncertainty informed by the stream of gathered observations. Nevertheless, the resulting inverse problem may be affected by biased estimations or even divergent solutions, and potentially become infeasible for stream applications.

Data-driven approaches rely on a *pattern recognition paradigm* [70] in which damage is assessed by comparing data previously obtained for known structural conditions with new incoming measurements. A pattern recognition paradigm for SHM was proposed in [71] and involves the following steps: (i) operational evaluation; (ii) data acquisition; (iii) *feature* selection and extraction; (iv) statistical modeling to unveil the relationship between the selected features and the sought damage patterns.

A feature is a synthetic representation of the data, ideally capable of encoding meaningful information for damage identification. Damage-sensitive features should be minimally affected by operational and environmental variabilities, making them suitable for damage identification purposes. *Feature selection* consists of determining which parts of the data are most sensitive to damage — a task that is complicated by the fact that data may be acquired from different sources with different sampling frequencies. This critical step is typically based on problem-specific knowledge subject to the available expertise. Often, features are related to the structural dynamic response. For instance, modal quantities obtained through operational modal analysis are a

63. Lye *et al.*, *A review of stochastic sampling methods for Bayesian inference problems*. 2019

64. Muto *et al.*, *Bayesian updating and model class selection for hysteretic structural models using stochastic simulation*. 2008

65. Gomes *et al.*, *A sunflower optimization (SFO) algorithm applied to damage identification on laminated composite plates*. 2019

66. Tran-Ngoc *et al.*, *Model Updating for Nam O Bridge Using Particle Swarm Optimization Algorithm and Genetic Algorithm*. 2018

67. Tiachacht *et al.*, *Damage assessment in structures using combination of a modified Cornwell indicator and genetic algorithm*. 2018

68. Ferrari *et al.*, *Model updating of a historic concrete bridge by sensitivity- and global optimization-based Latin Hypercube Sampling*. 2019

*The inherent ill-conditioning of the inverse calibration of a structural model refers to the lack of convexity in the associated optimization problem. Available solutions in the literature include the use of global optimization algorithms, regularization techniques, and probabilistic Bayesian strategies. An equally crucial aspect is the choice of an adequate parametrization for the forward model, typically determined through sensitivity analysis.

69. Kabanikhin, *Definitions and examples of inverse and ill-posed problems*. 2008

70. Bishop, *Pattern Recognition and Machine Learning*. 2006

71. Farrar *et al.*, *Vibration–Based Structural Damage Identification*. 2001

typical example of such features [72]. However, in several cases, features might not have a clear physical meaning, such as when they are extracted through crude statistical methods [73]. More generally, features extracted through data-based models may lack clear *interpretability*. For example, in the case of parameters and residuals of autoregressive models, as seen in [74,75].

The key difference between model-based and data-driven methods is the way in which features are exploited. Besides exploiting a physics-based model for feature selection, model-based approaches use data features to assess damage via model updating. On the other hand, data-driven approaches exploit the features to set up a *statistical model* describing a mapping between input features and output damage states, termed *labels*.

The adoption of the data-driven paradigm involves offline and online phases. During the *offline phase*, existing knowledge in the form of data, typically referring to undamaged or possibly damaged states of the structure, is exploited to *learn*† a function that describes the relationship between data features and damage states through a *training process*. During the *online phase*, the learned model is applied to incoming data to provide *informed predictions* of the structural health. Running incoming data through this model can thus be viewed as comparing their content with that of the training data. This offline-online decomposition typically allows faster predictions than those provided through model-based approaches. Other advantages of data-driven approaches over model-based ones include their capability to handle large amounts of raw noisy data and their flexibility, given the lack of any physics-based constraints. In this respect, data-driven approaches automatically learn relation between features and damage conditions during training.

The choice of the specific data-driven algorithm is influenced not only by the type of selected features but also by the specific task of damage identification to be addressed. The learning problem underlying damage identification can be categorized as either *supervised* or *unsupervised*. *Supervised methods* exploit training data corresponding to the baseline undamaged condition, as well as data pertaining to known damage scenarios that may affect the structure. In contrast, *unsupervised methods* rely solely on unlabeled training data, referring to the baseline condition.

When applied in unsupervised learning mode, statistical models are typically employed to address questions regarding the presence, and sometimes the location, of damage. Unsupervised methods are the primary choice for addressing the damage detection task by framing it as a novelty or anomaly detection problem. Popular techniques include control charts

72. W. Doebling *et al.*, *A Summary Review of Vibration–Based Damage Identification Methods*. 1998

73. Buckley *et al.*, *A Feature Extraction & Selection Benchmark for Structural Health Monitoring*. 2023

74. Wang *et al.*, *FEM Calibrated ARMAX Model Updating Method for Time Domain Damage Identification*. 2013

75. Entezami *et al.*, *An unsupervised learning approach by novel damage indices in structural health monitoring for damage localization and quantification*. 2018

†In this context, "learning" may refer to various strategies for establishing a statistical model based on training data. In the framework of statistical learning theory, the general case is formally addressed by selecting the mapping that minimizes the empirical risk associated with the training data. This typically reduces to tuning the weights ruling the employed statistical model by solving an optimization problem.

[76], k-nearest neighbor [77], statistics distance tests [78,79,80], outlier ensembles [81], isolation forests [82], kernel-based algorithms [83], as well as one-class support vector machines [84] and clustering algorithms [85], among others. Only training data from the undamaged condition of the structure are used to establish the diagnostic. This involves learning intrinsic relationships within the data to construct a model for the undamaged condition. During the monitoring stage, newly acquired data are compared for consistency with the model. If there are any significant deviations associated with the incoming data, the algorithm indicates novelty, using distance thresholds, decision boundaries, or densities, depending on the adopted algorithm‡. Similarly, unsupervised damage localization is typically conducted through an extended outlier analysis, exploiting knowledge about the positions of sensors to gain insights into the impact of the damage location on the structural response at various positions in the structure. Relevant algorithms include, e.g., the one-class support vector machine [86,87], and density-based and distance-based techniques [88,89].

Unsupervised solutions are sometimes preferred over supervised ones, as they only require data relevant to the baseline condition, without the need for a large amount of data to cover the range of variations in potential operational and damage conditions. However, in accordance with Axiom III of SHM [90] — with these Axioms attempting to provide a collection of general principles defining an SHM strategy — the identification of the type of damage, its severity, and often its location, generally requires a supervised learning approach.

In supervised learning, the statistical model is designed by establishing a mapping between the input data features and target labels within the training data. This is usually done by modeling the PDF of target labels conditioned on the input data features. Nevertheless, experimental data related to possible damage conditions of the structure are, a-priori, not available. To address this limitation, physics-based models can be coupled with the statistical pattern recognition in a supervised learning mode. Such a hybridization between data and physics can be thought of as a way to let data-driven models learn in a simulated environment before deployment in the real world [91].

Supervised damage localization can be formalized using either *classification* or *regression* methods. When approached from a classification perspective, query data features are categorized into an output class identifying a damage region among a set of predefined damage scenarios, each referring to a different damage location. Contributions in this domain include sup-

76. Giglioni *et al.*, *The use of receiver operating characteristic curves and precision-versus-recall curves as performance metrics in unsupervised structural damage classification under changing environment*. 2021

77. Entezami *et al.*, *Big data analytics and structural health monitoring: A statistical pattern recognition-based approach*. 2020

78. Entezami *et al.*, *Early damage assessment in large-scale structures by innovative statistical pattern recognition methods based on time series modeling and novelty detection*. 2020

79. Yan *et al.*, *Structural anomaly detection based on probabilistic distance measures of transmissibility function and statistical threshold selection scheme*. 2022

80. Worden *et al.*, *Damage detection using outlier analysis*. 2000

81. Bull *et al.*, *Outlier ensembles: A robust method for damage detection and unsupervised feature extraction from high-dimensional data*. 2019

82. Liu *et al.*, *Isolation Forest*. 2008

83. Santos *et al.*, *Machine learning algorithms for damage detection: Kernel-based approaches*. 2016

84. Goldstein *et al.*, *A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data*. 2016

85. Diez *et al.*, *A clustering approach for structural health monitoring on bridges*. 2016

‡The definition of the classification threshold (damage or non-damaged) typically poses a tricky issue, potentially leading to false positives and false negatives. The former, also known as false alarms, may result in an increase in inspection costs, while the latter, also known as missed alarms, may lead to unpredicted structural failures.

86. Wang *et al.*, *Automated damage-sensitive feature extraction using unsupervised convolutional neural networks*. 2018

87. Bigoni *et al.*, *Simulation-based Anomaly Detection and Damage Localization: An application to Structural Health Monitoring*. 2020

88. Cha *et al.*, *Unsupervised novelty detection-based structural damage localization using a density peaks-based fast clustering algorithm*. 2018

port vector machines and k-nearest neighbor [92], decision trees [45,93], and DL-based classifiers [94,95], among others. In these frameworks, the *target classes* related to potential damage scenarios the structure may experience must be defined in advance, based on prior knowledge about the mechanical behavior of the structure, the loading conditions, the load-bearing capacity of the structural members, the aging processes affecting similar structures, and on some prioritization study analyzing the occurrence probability of the various damage scenarios versus the relative consequences. Similarly, when damage quantification is approached from a classification perspective, the data features are used to categorize the damage severity, for instance, through support vector machines classifiers [96] or DL-based classifiers [97]. These concepts can be extended to the continuous case if damage localization and quantification are approached from a regression perspective. In this case, target quantities are predicted in a non-discretized setting. For damage localization, the data features are typically mapped onto a set of parameters describing the coordinates of a damage region. Similarly, for damage quantification, data features can be mapped onto the estimated damage severity. This approach intuitively avoids the a-priori definition of target classes, enabling a more precise description of the damage. Notable contributions in this field have been obtained by means of DL-based regressors [98,99,100] and support vector regressors [101], among others.

Feature selection/extraction and statistical modeling are typically considered as two separate steps in data-driven approaches. This aspect has motivated a continuous increase in the number of contributions leveraging on DL algorithms [102,103] in place of traditional machine learning (ML) methods. DL models for SHM purposes aim to automate the feature engineering stage, without requiring problem-specific knowledge for the given task. By capturing *temporal correlations* within and across time recordings, DL models enables the selection and extraction of damage-sensitive features from raw sensor recordings through an end-to-end learning process [104]. Optimized damage-sensitive features are selected and extracted to address the subsequent statistical modeling task. The DL paradigm also enables the automatic integration of data coming from different sources, such as temperature and structural acceleration recordings [105,106], offering clear advantages for the SHM of large structures, especially in presence of varying environmental and operational conditions [107,108]. Moreover, since the DL model is learned offline, the structural state can be assessed in real time regardless of considering continuous or discrete settings.

89. Entezami *et al.*, *Fast unsupervised learning methods for structural health monitoring with large vibration data from dense sensor networks*. 2020

90. Worden *et al.*, *The Fundamental Axioms of Structural Health Monitoring*. 2007

91. Yuan *et al.*, *Machine learning for structural health monitoring: challenges and opportunities*. 2020

92. Ritto *et al.*, *Digital twin, physics-based model, and machine learning applied to damage detection in structures*. 2021

45. Mariniello *et al.*, *Structural damage detection and localization using decision tree ensemble and vibration data*. 2021

93. Kapteyn *et al.*, *Toward predictive digital twins via component-based reduced-order models and interpretable machine learning*

94. Rosafalco *et al.*, *Online structural health monitoring by model order reduction and deep learning algorithms*. 2021

95. Abdeljaber *et al.*, *Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks*. 2017

96. Ghiasi *et al.*, *A machine-learning approach for structural damage detection using least square support vector machine based on a new combinational kernel function*. 2016

97. Sajedi *et al.*, *Vibration-based semantic damage segmentation for large-scale structural health monitoring*. 2020

98. Gulgec *et al.*, *Convolutional Neural Network Approach for Robust Structural Damage Detection and Localization*. 2019

99. Fernandez-Navamuel *et al.*, *Bridge damage identification under varying environmental and operational conditions combining Deep Learning and numerical simulations*. 2023

100. Rosafalco *et al.*, *A Self-adaptive Hybrid Model/data-Driven Approach to SHM Based on Model Order Reduction and Deep Learning*. 2022

101. Li *et al.*, *Damage Degree Identification of Crane Girder Based on the Support Vector Machine*. 2018

102. LeCun *et al.*, *Deep learning*. 2015

103. Goodfellow *et al.*, *Deep Learning*. 2016

1.2.3 OPERATIONAL EVALUATION

The *operational evaluation* process aims to address fundamental questions arising when implementing a damage identification strategy, such as defining the damage scenarios of interest and identifying the operational and environmental conditions in which the monitored system functions. Tailoring damage identification in this way ensures alignment with the unique characteristics of the monitored system [38].

104. Fink *et al.*, *Potential, Challenges and Future Directions for Deep Learning in Prognostics and Health Management Applications*. 2020

105. Torzoni *et al.*, *A Combined Model-Order Reduction and Deep Learning Approach for Structural Health Monitoring under Varying Operational and Environmental Conditions*. 2020

106. Torzoni *et al.*, *SHM under varying environmental conditions: An approach based on model order reduction and deep learning*. 2022

107. Alampalli, *Influence of in–service environment on modal parameters*. 1998

108. Sohn *et al.*, *Statistical Damage Classification Under Changing Environmental and Operational Conditions*. 2002

38. Farrar *et al.*, *Structural Health Monitoring: A Machine Learning Perspective*. 2013

The definition of damage encompasses considerations such as the specific type of damage to be detected, the threshold level of damage that needs detection, and the locations within the structure where damage tends to accumulate. These definitions often rely on the observed behavior of previously damaged systems. Numerical simulations of the system are also commonly employed to establish these definitions. As commented in Sec. 1.2.2, the identification of potential damage scenarios of interest should also involve a prioritization study that analyzes the occurrence probability of various damage scenarios compared to their relative consequences. This prioritization then influences the choice of the data acquisition system and the candidate features to be extracted for damage identification.

Environmental and operational variations (EOVs) may introduce changes that can substantially impact the measured dynamic quantities, hiding damage effects and potentially spoiling the effectiveness of a damage identification technique. It is crucial to ensure that these changes are not misinterpreted as indications of damage. In this light, the need for operational evaluation is well captured by Axiom IVb of SHM [90]:

90. Worden *et al.*, *The Fundamental Axioms of Structural Health Monitoring*. 2007

> "Without intelligent feature extraction, the more sensitive a measurement is to damage, the more sensitive it is to changing operational and environmental conditions."

All real systems are subject to measurement noise and typically operate in a changing environment. The SHM system must be able to distinguish between a statistical fluctuation in the data and a real deviation from normality*. Clearly defining and quantifying the spectrum of operational and environmental conditions is crucial for refining the capabilities required for the data acquisition system. In many cases, additional sensors may be necessary to monitor evolving operational and environmental conditions, enabling procedures that normalize the data by eliminating EOVs-induced trends.

*The term "normal operating condition" refers to a state of the system that is fit-for-purpose.

The challenge of confounding influences is particularly pronounced in civil infrastructures. In the case of bridges, for instance, the interaction between the structure and moving vehicles alters the dynamic characteristics, depending on factors

like vehicle velocity and traffic mass. Fluctuating temperature and moisture levels are two prevalent factors that often hamper the correct assessment of the damage state in civil structures. Daily and seasonal thermal fluctuations notably impact material properties and may alter the boundary conditions of the system. Temperature gradients resulting from solar radiation can influence the vibration response of a structure, causing changes in its structural frequencies and mode shapes, as well as inducing geometric distortions [109]. Additionally, minor damage events in civil structures can induce changes in the structural response that are often of limited relevance and potentially challenging to be distinguished from measurement noise and EOVs.

109. Farrar *et al.*, *Variability of modal parameters measured on the Alamosa Canyon Bridge*. 1996

Data normalization techniques are essential for mitigating EOVs effects, in order to relate changes in data content to the onset and propagation of damage. When environmental measurements are available, they can be used to characterize the dependence of the (observable) effects of damage on the operational conditions. A possible approach of this type is to construct a data-based response surface model capable of representing measurement variations as a function of environmental variables to get rid of their effects [110,111]. Alternatively, a linear filter can be used to set the structural frequencies as a function of temperature readings, thus leading to damage-sensitive and environment-insensitive features [112]. Blind approaches can instead prove useful in scenarios where environmental measurements are unavailable. This is the case of algorithms that can project out the environmentally-sensitive components of data features while preserving the damage-sensitive ones. For instance, principal component analysis can remove the influence of temperature fluctuations on the measurements by projecting data onto their principal components associated with smaller information content, assumed to be insensitive to environmental factors [113]. Autoregressive models with exogenous inputs based on auto-associative neural networks have been also employed in order to code the environmental and operational conditions as *latent variables* [108]. Finally, cointegration is a recent approach originating in econometric that enables establishing combinations of features purged from common trends induced by the environmental conditions [114,115].

110. Worden *et al.*, *On switching response surface models, with applications to the structural health monitoring of bridges*. 2018

111. Cross *et al.*, *Long-term monitoring and data analysis of the Tamar Bridge*. 2013

112. Sohn *et al.*, *An experimental study of temperature effect on modal parameters of the Alamosa Canyon Bridge*. 1999

113. Manson, *Identifying damage sensitive, environment insensitive features for damage detection*. 2002

108. Sohn *et al.*, *Statistical Damage Classification Under Changing Environmental and Operational Conditions*. 2002

114. Cross *et al.*, *Features for damage detection with insensitivity to environmental and operational variations*. 2012

115. Coletta *et al.*, *Use of the cointegration strategies to remove environmental effects from data acquired on historical buildings*. 2019

Unfortunately, all these approaches need historical data recorded across a full spectrum of environmental and operational conditions. To overcome this requirement, a physics-based model of the structure capable of capturing the thermo-elastic coupling may overcome this challenge by directly incorporating the EOVs effects through multi-physics descriptions. In line

with the simulation-based paradigm of SHM, physics-based models should facilitate the simulation of thermo-mechanical effects arising from material contraction/expansion and stiffening/softening, to collect vibration and thermal data under different operational and environmental conditions [105,106].

105. Torzoni *et al.*, *A Combined Model-Order Reduction and Deep Learning Approach for Structural Health Monitoring under Varying Operational and Environmental Conditions*. 2020

106. Torzoni *et al.*, *SHM under varying environmental conditions: An approach based on model order reduction and deep learning*. 2022

1.2.4 Coping with the lack of data

The simulation-based paradigm of SHM, as introduced in Sec. 1.2.1, can significantly enhance damage identification strategies beyond simple damage detection. However, while physics-based numerical models can help reproduce potential damage scenarios, they may suffer from representability issues, leading to diverging distributions between simulated and experimental data. Simulated data and experimental data typically follow different distributions, posing a risk to the *generalization capabilities* of damage identification techniques. Methodologies well-trained in a simulated environment may experience a significant drop in performance when applied to the target structure. Moreover, implementing case-specific algorithms for each structure with an installed SHM system is impractical.

To address these issues, *population-based SHM* [116,117,118] is a promising field aiming to gather comprehensive damage-level information from similar assets. In this case, the target application is viewed as a subset of a larger domain, comprising damage scenarios for a population of similar structures. However, the distribution underlying damage-sensitive features* may differ across structures due to variations in design, materials, geometry, sensor type, or operational conditions. To address this challenge, *transfer learning* has emerged in SHM as a way to leverage damage-state knowledge gained from a well-understood *source domain* and infer diagnoses on a different, typically less-explored or changing, *target domain*.

116. Bull *et al.*, *Foundations of population-based SHM, Part I: Homogeneous populations and forms*. 2021

117. Gosliga *et al.*, *Foundations of Population-based SHM, Part II: Heterogeneous populations – Graphs, networks, and communities*. 2021

118. Gardner *et al.*, *Foundations of population-based SHM, Part III: Heterogeneous populations – Mapping and transfer*. 2021

*Raw experimental data are unsuitable to be treated as random variables due to intrinsic difficulties related to their dimensionality. Instead, a probabilistic perspective can be adopted for damage-sensitive features extracted from the raw data.

Domain adaptation is a subfield of transfer learning where the goal is to minimize the distance between data distributions associated with source and target domains within a shared feature space. A substantial portion of recent literature on *domain adaptation* strategies focuses on generative adversarial learning. In *adversarial domain adaptation*, as observed in [119,120], a feature extractor and a domain discriminator are trained adversarially, yielding features extracted by the generator that are shared by both the source and the target domains. Another approach, developed in works such as [48,121,122], is *statistical alignment*, aiming to align low-order statistics of source and target domains in a sheared feature space.

119. Tzeng *et al.*, *Adversarial Discriminative Domain Adaptation*. 2017

120. Wang *et al.*, *Knowledge transfer for structural damage detection through re-weighted adversarial domain adaptation*. 2022

48. Giglioni *et al.*, *A domain adaptation approach to damage classification with an application to bridge monitoring*. 2024

121. Gardner *et al.*, *On the application of domain adaptation in structural health monitoring*. 2020

122. Poole *et al.*, *On statistic alignment for domain adaptation in structural health monitoring*. 2023

1.3 THE NEED FOR REDUCED-ORDER AND SURROGATE MODELS

In Sec. 1.2.1, we have discussed the challenge of obtaining and accumulating knowledge on the structural response of civil structure in the presence of varying damage conditions. The simulation-based paradigm of SHM has been introduced to enable a knowledge platform where the effect of damage on the structural response can be systematically reproduced using physics-based models. This simulation-based knowledge can take the form of a forward model, in the case of model-based approaches, or in a database that collects data pertaining to specific damage conditions, in the case of data-driven approaches.

1.3.1 MANY-QUERY APPLICATIONS

The simulation-based paradigm of SHM offers an opportunity to extend damage identification strategies beyond simple damage detection through simulation-based supervision. However, model-based approaches can incur high computational costs due to the repeated evaluation of computationally intensive forward models, with the risk of preventing real-time applications. Similarly, data-driven approaches may demand extensive computational efforts because of the need to simulate and accumulate structural response data for various combinations of damage, operational, and (possibly) environmental conditions*.

*Considerations regarding the computational model chosen to populate a training datasets and its associated computational cost are particularly intricate, involving a delicate trade-off between the quantity/quality of the training data and the performance of the data-driven model to be learned. For instance, in the case of highly parametrized models such as deep learning models, the number of required training data points may be extremely high.

These two scenarios represent instances of *many-query applications*. This term refers to applications where a model must be evaluated many times. A subset of many-query applications is that of *outer-loop applications*, which are characterized by targeting a specific outer-loop result. In contrast, many-query applications do not necessarily aim for a specific outer-loop result. In the context of Bayesian model inversion, the task of determining the values of parameters to best fit model predictions to the observations can be regarded as an outer-loop application. Differently, the process of populating a large dataset to train a data-driven model can be viewed as a many-query application that is not an outer-loop one. In these cases, the computational cost of employing a high-fidelity (HF) model often becomes prohibitive. Introducing a cheaper low-fidelity (LF) model can be crucial to allow for a high number of evaluations with (possibly) slightly reduced accuracy [123]. Replacing computationally demanding numerical models with cost-effective LF models can reduce the computational burden when populating datasets for data-driven approaches or when feeding outer-loop optimization algorithms in the case of model-based approaches.

123. Peherstorfer *et al.*, *Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization*. 2018

Apart from employing simplified physics or a coarser discretization (either in time or space), notable instances of LF models have been achieved through *model order reduction* strategies aimed at reducing the number of degrees of freedom (DOFs) of a HF computational model. Popular strategies for approximating an HF full-order model (FOM) with a cheaper, yet accurate, LF reduced-order model (ROM) include reduced basis methods based on proper orthogonal decomposition or greedy strategies [124,125], proper generalized decomposition [126,127], balanced truncation and Hankel norm approximation methods [128], Krylov methods such as Lanczos [129] and Arnoldi [130], spectral proper orthogonal decomposition [131,132], Grassmannian diffusion maps [133], artificial neural networks [134], and operator inference [135,136,137], among the others. From a different perspective, additional instances of LF models have been obtained by relying on *surrogate modeling* techniques, designed to construct a data-driven emulator that mimics the HF input-output behavior[†]. Popular strategies for building surrogate models (SMs) include polynomial response surface models, polynomial chaos, support vector machines, Kriging models, and artificial neural networks [138,139,140,141,142,143,144].

1.3.2 Multi-fidelity methods

An intriguing opportunity to further reduce the computational cost of numerical methods may arise in situations where multiple models are available to simulate the relevant structural response with varying accuracy and computational costs. This is the case of multi-fidelity (MF) methods, which combine LF with HF models to achieve improved approximation accuracy compared to the LF solution while maintaining a lower computational burden than the HF solver. Indeed, LF samples often provide useful information on the major trends of the problem, allowing the MF setting to outperform single-fidelity methods in terms of prediction accuracy and computational efficiency.

Motivated by recent advances in MF methods, the use of DL models to enhance MF approaches represents a recent paradigm in scientific ML that has been developed, for instance, in [145,146,147,148,149,150]. The premise for this setup is that the correlation across structural models featuring different fidelity levels can be leveraged without any prior assumption, thanks to the flexibility of neural networks. The resulting framework typically relies on a composition of deep neural network (DNN) models and is therefore termed MF-DNN. This ultimately alleviates the computational burden of supervised training while ensuring the accuracy of the approximated quantities of interest.

124. Kerschen *et al.*, *The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview*. 2005

125. Quarteroni *et al.*, *Reduced Basis Methods for Partial Differential Equations: An Introduction*. 2015

126. Ladevèze, *Nonlinear Computational Structural Mechanics: New Approaches and Non-Incremental Methods of Calculation*. 1999

127. Chinesta *et al.*, *A Short Review in Model Order Reduction Based on Proper Generalized Decomposition*. 2011

128. Antoulas, *Approximation of Large-Scale Dynamical Systems*. 2005

129. Lanczos, *An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators*. 1950

130. Arnoldi, *The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem*. 1951

131. Sieber *et al.*, *Spectral proper orthogonal decomposition*. 2016

132. Towne *et al.*, *Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis*. 2018

133. Santos *et al.*, *Grassmannian diffusion maps based surrogate modeling via geometric harmonics*. 2022

134. Fresca *et al.*, *Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Nonlinear Time-Dependent Parametrized PDEs*. 2021

135. Peherstorfer *et al.*, *Data-driven operator inference for non-intrusive projection-based model reduction*. 2016

136. Qian *et al.*, *Reduced Operator Inference for Nonlinear Partial Differential Equations*. 2022

137. Geelen *et al.*, *Operator inference for non-intrusive model reduction with quadratic manifolds*. 2023

[†]The "input-output behavior" refers to the pipeline of drawing samples from a parametric space, and then evaluating a forward computational model to assess the corresponding solution.

138. Forrester *et al.*, *Engineering Design Via Surrogate Modelling: A Practical Guide*. 2008

1.4 THESIS OBJECTIVES

The primary objective of this Thesis is to advance computational techniques for enabling predictive DTs for structural systems, leveraging state-of-the-art methodologies from data science and engineering. This involves introducing novel perspectives in both model-based and data-driven approaches to SHM by hybridizing physics-based modeling with *artificial intelligence*.

The original contributions of this Thesis are:

I **A damage localization framework exploiting deep metric learning and simulation-based supervision. (Chapter 5)**

II **A multi-fidelity, deep neural network surrogate modeling strategy for SHM purposes. (Chapter 6)**

III **A strategy to enhance Bayesian model updating using learnable mappings. (Chapter 7)**

IV **A computational framework for predictive digital twins of engineering structures. (Chapter 8)**

These contributions have been motivated by several factors and aim to address specific challenges in the existing literature. **Contribution I** aims to enhance the accuracy and interpretability of DL-based strategies for damage identification by leveraging deep metric learning for improved feature extraction and representation. **Contribution II** addresses challenges related to computational efficiency and scalability of Bayesian approaches to SHM by blending MF modeling and DNNs to create accurate and versatile SMs. **Contribution III** aims to provide an effective approach to Bayesian model updating by capturing complex relationships in the data via learnable mappings. **Contribution IV** aims to leverage SHM strategies within a DT framework, demonstrating the effectiveness of diagnostic and predictive elements to realize value through proactive decision-making.

Despite targeting different objectives, the four contributions share the same underlying spirit of moving towards reliable, possibly real-time, early identification of damage patterns in civil structures. This challenge is addressed through the joint use of simulation-based supervision, to overcome the unavailability of data related to the onset and propagation of damage, and learnable mappings, to handle and process high-dimensional data, enhance damage identification capabilities, and improve computational efficiency. In the following, we briefly comment on the key aspects of these contributions in terms of methodological development and obtained results.

139. Koziel *et al.*, *Surrogate-Based Modeling and Optimization: Applications in Engineering*. 2013

140. Maulik *et al.*, *Probabilistic neural networks for fluid flow surrogate modeling and data recovery*. 2020

141. Sadeghi Eshkevari *et al.*, *DynNet: Physics-based neural architecture design for nonlinear structural response modeling and prediction*. 2021

142. García-Macías *et al.*, *An Innovative Methodology for Online Surrogate-Based Model Updating of Historic Buildings Using Monitoring Data*. 2021

143. Raissi *et al.*, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*. 2019

144. Gunn, *Support Vector Machines for Classification and Regression*. 1998

145. Geneva *et al.*, *Multi-fidelity generative deep learning turbulent flows*. 2020

146. Zhang *et al.*, *Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization*. 2021

147. Meng *et al.*, *A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems*. 2020

148. Meng *et al.*, *Multi-fidelity Bayesian Neural Networks: Algorithms and Applications*. 2021

149. Guo *et al.*, *Multi-fidelity regression using artificial neural networks: Efficient approximation of parameter-dependent output quantities*. 2022

150. Conti *et al.*, *Multi-fidelity surrogate modeling using long short-term memory networks*. 2023

A DAMAGE LOCALIZATION FRAMEWORK EXPLOITING DEEP METRIC LEARNING AND SIMULATION-BASED SUPERVISION.

The first contribution of this Thesis is the development of a data-based, supervised, and deterministic diagnostic framework for real-time damage localization. This is achieved by combining some aspects from *metric learning* and *dimensionality reduction* techniques. Metric learning [151,152,153] aims to acquire a distance function that adheres to a task-specific definition of *similarity*, ensuring that neighbors of a data point are mapped closer than non-neighbors in the learned metric space. Recognizing the challenge of directly inferring quantities of interest from sensor data, due to the *curse of dimensionality*, dimensionality reduction is employed to map high-dimensional vibration recordings onto a *low-dimensional manifold*, embedding the relevant semantic content* [154,155]. In concrete terms, this strategy leverages a learnable mapping of raw vibration measurements onto a low-dimensional metric space, wherein structural health parameters describing damage locations can be easily discriminated. The mapping is learned in a supervised pairwise fashion through a *Siamese* [156] *convolutional architecture* trained on labeled data from ROM simulations. The Siamese architecture allows for establishing a feature space equipped with a task-specific metric (refer to Fig. 1.3). During the prediction stage for previously unseen data, a simple regressor is applied in the learned low-dimensional feature space, providing real-time damage localization.

151. Kaya *et al.*, *Deep Metric Learning: A Survey*. 2019

152. Bellet *et al.*, *A Survey on Metric Learning for Feature Vectors and Structured Data*. 2013

153. Cakir *et al.*, *Deep Metric Learning to Rank*. 2019

*The term "semantic" stems from the fact that the components of the low-dimensional space encode the position of the damage in the structure, which represents the associated semantic content.

154. Hadsell *et al.*, *Dimensionality Reduction by Learning an Invariant Mapping*. 2006

155. Chopra *et al.*, *Learning a similarity metric discriminatively, with application to face verification*. 2005

156. Bromley *et al.*, *Signature Verification using a "Siamese" Time Delay Neural Network*. 1993

The strength of this procedure lies in the supervised pairwise training, allowing for a damage-sensitive manifold capable of representing distances in the physical space, yielding easily interpretable results. Additionally, by capturing temporal correlations within and across time recordings, the use of DL models with convolutional layers enables the automatic selection and extraction of damage-sensitive features from raw vibration recordings. Since damage is assessed from a regression perspective, this strategy eliminates the need of defining target classes in advance. It also avoids tuning highly sensitive *hyperparameters*, such as distance thresholds and decision boundaries.

The methodology is investigated by employing three alternative loss functions to train the Siamese network and varying the size of the low-dimensional feature space. Simulation results, considering noise-contaminated measurements, demonstrate that this strategy exhibits remarkable damage localization capabilities, outperforming an alternative real-time method.

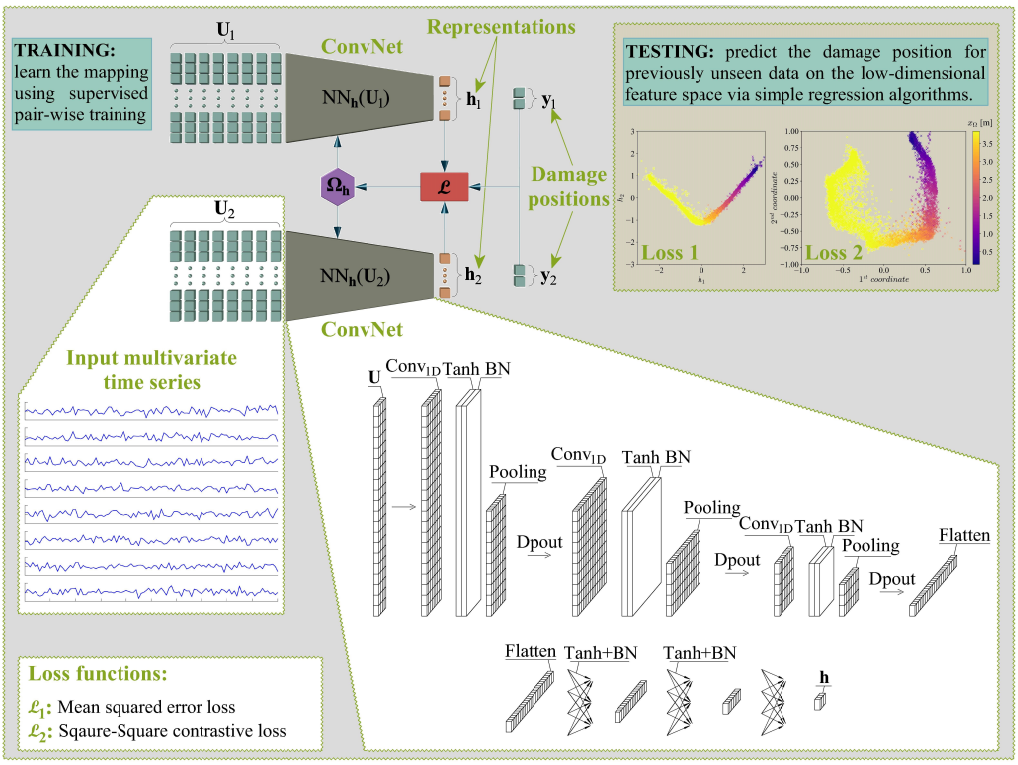


Figure 1.3: Damage localization framework exploiting deep metric learning: graphical abstraction.

A MULTI-FIDELITY, DEEP NEURAL NETWORK SURROGATE MODELING STRATEGY FOR SHM PURPOSES.

The second contribution of this Thesis introduces a *non-intrusive* MF surrogate modeling strategy designed to approximate sensor recordings under varying damage and operational conditions. This strategy exploits model order reduction and DNNs to reduce the computational cost of supervised training while ensuring the accuracy of the approximated signals in terms of damage sensitivity. The resulting surrogate (MF-DNN-SM) features a multi-level architecture of sequentially trained DNNs, comprising both an LF and an HF part. The LF-DNN is a *fully-connected model* employed to mimic sensor recordings in the undamaged condition, while the HF-DNN is a *long short-term memory model* used to adaptively refine the LF approximation with the effect of damage. The training involves using a small HF dataset and a large LF dataset (refer to Fig. 1.4). Vibration response data required for fitting the MF surrogate are generated using an LF-ROM obtained through the reduced basis method and an HF-FE model. These models simulate the structural response under varying operational conditions. In particular, the LF model refers to the baseline undamaged condition, while the HF model accounts for potential degradation processes.

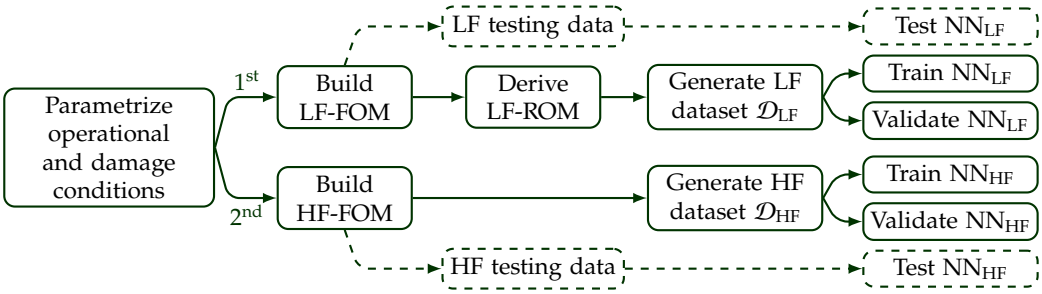


Figure 1.4: Multi-fidelity deep neural network surrogate model: methodology flowchart.

The novelty of the proposed methodology lies in the tailored adaptation of the MF-DNN surrogate modeling framework for SHM purposes. The designed MF-DNN-SM is specifically engineered to map damage and operational parameters onto sensor recordings, a challenging task for other MF techniques like *Gaussian processes-based Cokriging* [157,158,159]. The computational complexity of Cokriging scales cubically with the total number of HF and LF training points, making it less suitable for high-dimensional problems enjoying large LF training datasets. Moreover, Cokriging models require prior knowledge about the correlation between fidelity levels and typically captures only linear correlations. In contrast, the MF-DNN-SM excels in various aspects: (i) it is well-suited for high-dimensional problems and benefits from large LF datasets; (ii) it learns both linear and nonlinear correlations adaptively; (iii) it provides real-time predictions for new input parameters; and (iii) it can handle the approximation of strongly discontinuous functions.

157. Kennedy *et al.*, *Predicting the Output from a Complex Computer Code When Fast Approximations Are Available.* 2000

158. Kast *et al.*, *A non-intrusive multifidelity method for the reduced order modeling of nonlinear problems.* 2020

159. Goh *et al.*, *Prediction and Computer Model Calibration Using Outputs From Multifidelity Simulators.* 2013

The MF-DNN-SM not only inherits the main features of MF methods, such as reduced offline computational burden and improved approximation accuracy, but also offers additional advantages: (i) the effect of damage on the structural response is simulated with the HF model alone, considered the most accurate description for potential damage scenarios; (ii) a large amount of training data relevant to the undamaged condition can be simulated, once and for all, exploiting the LF-ROM to speed up the offline training phase; and (iii) there is no need to update the LF part, since when a deterioration in structural health is detected, the MF surrogate can be updated by adjusting only its HF component.

The obtained results demonstrate the efficiency of this procedure in providing remarkably accurate approximations, outperforming its single-fidelity counterpart. The capability of the MF-DNN-SM to be exploited for SHM purposes is finally showcased within an automated Bayesian procedure, accounting for operational variability and measurement noise.

A STRATEGY TO ENHANCE BAYESIAN MODEL UPDATING USING LEARNABLE MAPPINGS

The third contribution of this Thesis is the development of a *hybrid* data-driven/model-based, supervised, stochastic strategy for real-time damage localization and quantification.

The selection/extraction of synthetic and informative damage-sensitive features from raw sensor recordings represents a critical step in the pattern recognition paradigm for SHM. In this regard, DL represents a promising solution to automate and optimize this process for the task at hand. On a different front, Bayesian model-based approaches to SHM assess damage from a parameter estimation perspective, employing a model updating strategy. This probabilistic framework has the advantage of naturally dealing with the ill-posedness of the SHM problem, allowing for the quantification of the associated uncertainty. This contribution introduces a hybrid approach to SHM, capitalizing on the strengths of both data-driven and model-based approaches to enhance stochastic approaches to SHM.

The proposed strategy leverages a learnable feature extractor and a feature-oriented SM, both based on DNNs. These two data-driven models work in tandem to evaluate a *likelihood function* within a *Markov chain Monte Carlo* (MCMC) sampling algorithm, updating beliefs about structural health parameters. Learnable features, optimized for the monitored structure, are automatically selected and extracted by the DL-based feature extractor. This maps the input vibration recordings onto their feature representation in a low-dimensional space, relying on an *autoencoder architecture* for dimensionality reduction purposes. During training, the autoencoder is equipped with a Siamese appendix of the encoder, optimized through a pairwise contrastive learning strategy [154,155]. Similar to the deep metric learning strategy introduced in Sec. 1.4, this approach enables encoding the sensitivity to the sought parameters in the learned metric space, thereby allowing a manifold to capture the parametric space describing the variability of the structural system. The extracted features are employed to evaluate the likelihood function of an MCMC sampler by means of a feature-oriented SM that approximates the functional link between the parameters to be updated and the feature space.

154. Hadsell *et al.*, *Dimensionality Reduction by Learning an Invariant Mapping*. 2006

155. Chopra *et al.*, *Learning a similarity metric discriminatively, with application to face verification*. 2005

Before feature extraction, the time series of sensor recordings are pre-processed and transformed into images. This process, known as *imaging time series*, enhances the visibility of local patterns in the data that might otherwise be spread over or lie outside the time domain. This encoding is achieved using the recently proposed *Markov transition field* (MTF) technique [160].

160. Wang *et al.*, *Imaging Time-Series to Improve Classification and Imputation*. 2015

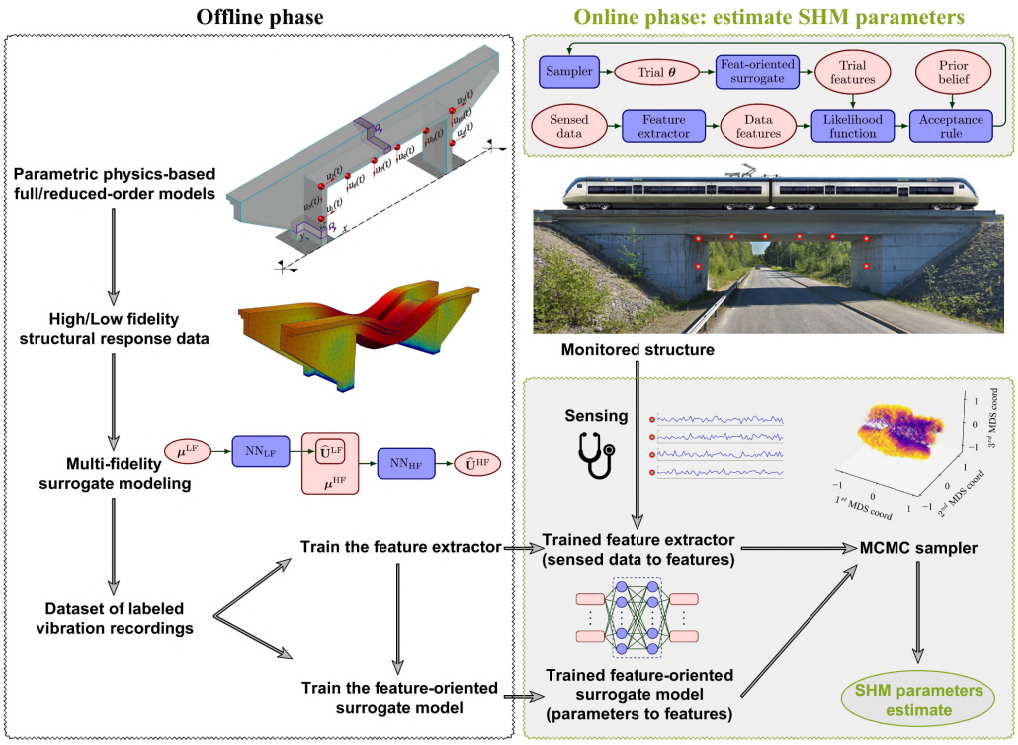


Figure 1.5: Enhanced Bayesian model updating: graphical abstraction.

A graphical abstraction of the considered framework is reported in Fig. 1.5. Vibration responses at different fidelity levels are simulated offline using physics-based full/reduced-order numerical models. These data are then exploited to train a MF-DNN-SM. Once trained, the MF-DNN-SM is employed to cost-effectively generate a significantly large training dataset. This dataset is used to train the feature extractor and the feature-oriented SM in a supervised fashion. During the online monitoring phase, these two DL models are exploited within an MCMC sampler to update the prior belief about the structural state.

In addition to assimilating vibration responses data through DL models, the main element of novelty lies in the employed low-dimensional feature space, which incorporates a geometric structure that encodes the sensitivity to the sought parameters. The resulting MCMC framework enjoys: (*i*) competitive computational costs due to the low dimensionality of the involved features; (*ii*) fast convergence thanks to the geometric structure characterizing the feature space; and (*iii*) accurate estimates resulting from the informativeness of the extracted features.

The obtained results show that using learnable features instead of vibration recordings leads to remarkable improvements in the accuracy and efficiency of parameter estimation.

A COMPUTATIONAL FRAMEWORK FOR PREDICTIVE DIGITAL TWINS OF ENGINEERING STRUCTURES

The final contribution of this Thesis involves the development of a DT framework for health monitoring, maintenance, and management planning of engineering structures. The computational strategy is based upon a probabilistic graphical model (PGM) inspired by the foundational model proposed in [19], which provides a general framework for data assimilation, state estimation, prediction, planning, and learning. Formally, this PGM is a *dynamic Bayesian network* with the addition of decision nodes, i.e., a *dynamic decision network* [161,162]. The PGM encodes the asset-twin dynamical system and its temporal evolution, governing the *observations-to-decisions flow*, from physical to digital through assimilation and inference, and back to the physical asset in the form of *informed control actions*, while quantifying the associated uncertainty. A graphical abstraction of the DT strategy is depicted in Fig. 1.6, illustrating both the *physical-to-digital* and *digital-to-physical* information flows[†]. This bi-directional interaction repeats indefinitely over time. In detail, we have:

19. Kapteyn *et al.*, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*. 2021

161. Koller *et al.*, *Probabilistic Graphical Models: Principles and Techniques*. 2009

162. Russell, *Artificial Intelligence: A Modern Approach*. 2020

[†]This comprehensive interaction among the physical system, data collection, computational models, and decision-making processes is rooted in the "Dynamic Data Driven Applications Systems" paradigm [163]. This is often considered the initial attempt to formalize the digital twin concept. It recognized the potential of incorporating observational data into an executing computational model to dynamically steer the measurement process through a feedback-and-control loop between simulations and measurements.

- *From physical to digital.* Observational data are collected from the physical asset and assimilated for structural health diagnostics using DL models. This process continually updates a digital state, describing the structural health of the asset, through *sequential Bayesian inference*.

- *From digital to physical.* The updated digital state is employed to forecast future damage growth based on *control-dependent state transition models*. These models describe how the structural health is expected to evolve, enabling predictive decision-making about maintenance and management actions feeding back to the physical system.

- *Offline learning phase.* This phase involves training the DL models for structural health identification, and learning the *control policy* to be applied at each time step during the online phase. The DL models are trained in a supervised fashion, using labeled data from ROMs simulations. The health-dependent control policy is also computed offline by maximizing the expected future rewards for the planning problem induced by the PGM.

The novel elements characterizing this contribution include: (i) the adaptation of the PGM digital twinning framework for the health monitoring and management planning of civil structures; (ii) the assimilation of vibration response data through DL

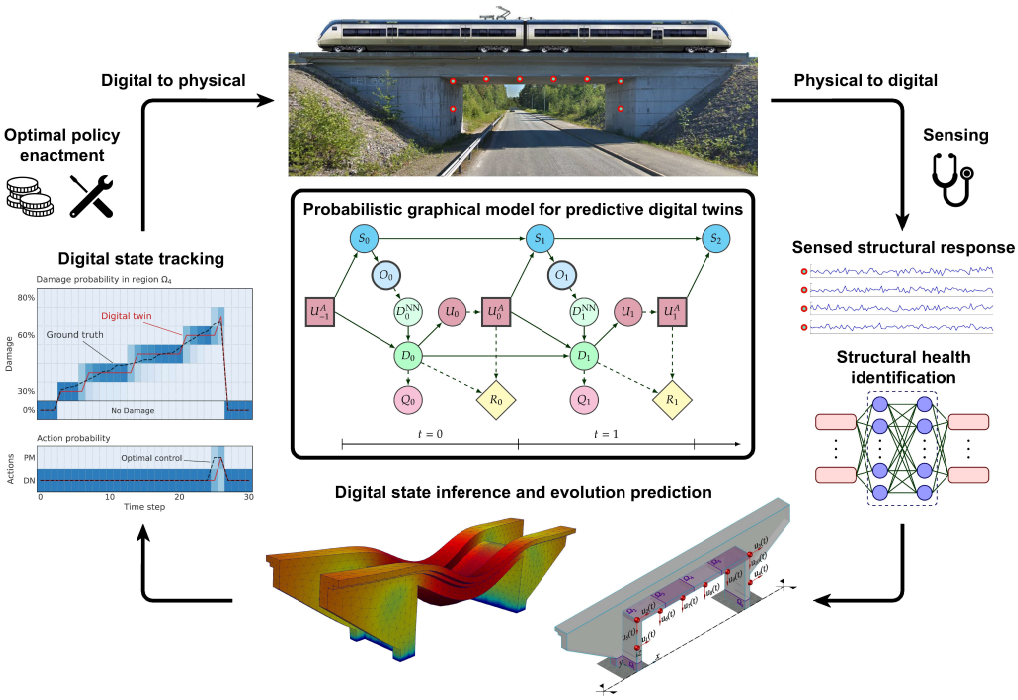


Figure 1.6: Predictive digital twin framework for civil engineering structures: graphical abstraction.

models, employing DL classifiers for damage detection and localization, and DL regressors for damage quantification.

The strategy is assessed through the simulated monitoring of different test cases, demonstrating the capabilities of health-aware DTs to track the evolution of structural health parameters and promptly suggest the most appropriate control input with relatively low uncertainty.

1.5 THESIS OUTLINE

The dissertation is organized as follows (see also Fig. 1.7 for a graphical overview).

Chapter 2 provides a methodological understanding of the differences underlying model-based and data-driven approaches to SHM, useful for identifying the key aspects and challenges associated with each approach. The model-based and data-driven dichotomy is vividly exemplified by reviewing two well-established methods for parameter identification, namely MCMC sampling and k-nearest neighbors regression.

In **Chapter 3**, we analyze the physics-based computational models employed in this Thesis. Because they allow us to address the pathological lack of experimental data related to damage conditions possibly affecting the structure to be monitored, these models are the key enablers for the development of damage

identification strategies within the simulation-based paradigm of SHM. Starting from the description of a reference full-order FE model, we then detail a reduced-order modeling strategy relying on a proper orthogonal decomposition reduced basis method for parametrized systems. We derive the parametrized ROM and show how it can systematically reproduce the structural response under varying operational and damage conditions, eventually allowing for the collection of synthetic vibration recordings mimicking a pervasive sensing system.

Chapter 4 delves into the analysis of several ML methods. We introduce the key aspects of ML, encompassing the rationale behind regression and classification algorithms, the supervised and unsupervised training of a learnable model, and the DL paradigm. This latter is extensively analyzed, describing several DL architectures and highlighting the peculiarities that make them suitable for SHM purposes. Finally, we consider the mathematical and statistical underpinnings of PGMs. The synergistic use of these methods is not only at the basis of the pattern recognition paradigm for SHM. More generally, this is envisioned to provide the algorithmic platform to support the physics-based modeling discussed in Chapter 3 for realizing effective DTs of engineering structures.

Chapter 5 is devoted to the development of the damage localization framework exploiting deep metric learning and simulation-based supervision (contribution I). The main focus of the Chapter is on learning a nonlinear mapping of raw vibration measurements onto a feature space where structural health parameters can be easily discriminated. We comment on the proposed strategy from the perspective of both metric learning and dimensionality reduction. We deeply discuss the characteristics of the convolutional DL model characterizing the proposed approach. The robustness and effectiveness of the proposed methodology are investigated through two different test cases by adopting three alternative loss functions and assessing the impact of critical modeling aspects on the damage localization performance.

Chapter 6 is devoted to adapting the MF surrogate modeling framework for SHM purposes using ROMs and DNNs (contribution II). We describe the numerical models considered for generating synthetic datasets of varying fidelity, as well as the composition of the learnable components underlying the MF-DNN-SM and their sequential training. The effectiveness and efficiency of the MF-DNN-SM are investigated by considering two test cases, and the obtained results are compared with single-fidelity approximations. Finally, we assess the possibil-

ity of exploiting the MF-DNN-SM in an MCMC-based model updating framework for SHM purposes.

Chapter 7 is devoted to the hybrid data-driven/model-based, supervised, stochastic strategy for real-time damage localization and quantification (contribution III). The Chapter focuses on exploiting the deep metric learning framework proposed in Chapter 5 to automate feature extraction via DL models, and the MF-DNN-SM achieved in Chapter 6 to generate large labeled datasets, thus reducing the cost of supervised training. We show how to pre-process training datasets of sensor recordings to generate image datasets via MTF encoding. We provide detailed insights into the architectures and the technical aspects related to training and evaluation of the feature extractor and the feature-oriented SM used within an MCMC-based parameter estimation framework. The effectiveness and reliability of the strategy are investigated by considering three different test cases of increasing structural complexity, demonstrating remarkable parameter identification outcomes in terms of computational cost, convergence rate, and estimation accuracy.

Chapter 8 is devoted to the computational framework for predictive digital twins of engineering structures (contribution IV). By blending the simulation-based paradigm of SHM and the DT concept, this Chapter provides a practical integration of several aspects addressed throughout this Thesis, from data-driven parameter identification and physics-based modeling, to PGMs and sequential decision making. We describe the dynamic decision network used to encode the asset-twin system and its evolution over time. We discuss the assimilation of vibration measurements with DL models trained on ROMs data, leveraging both classification and regression perspectives to assess the structural health. We combine the DL models' predictions with state transition probabilities to update the DT via sequential Bayesian inference. Furthermore, we show how to forecast future trajectories of structural health and inform predictive decision-making about maintenance and management actions. We assess the dynamic decision-making capabilities of health-aware DTs through simulated demonstrations of dynamic data-driven health monitoring, prediction, and planning.

Finally, **Chapter 9** draws the conclusions of this Thesis, summarizes the outcomes, discusses their limitations, and envisions the outlook for possible future developments.

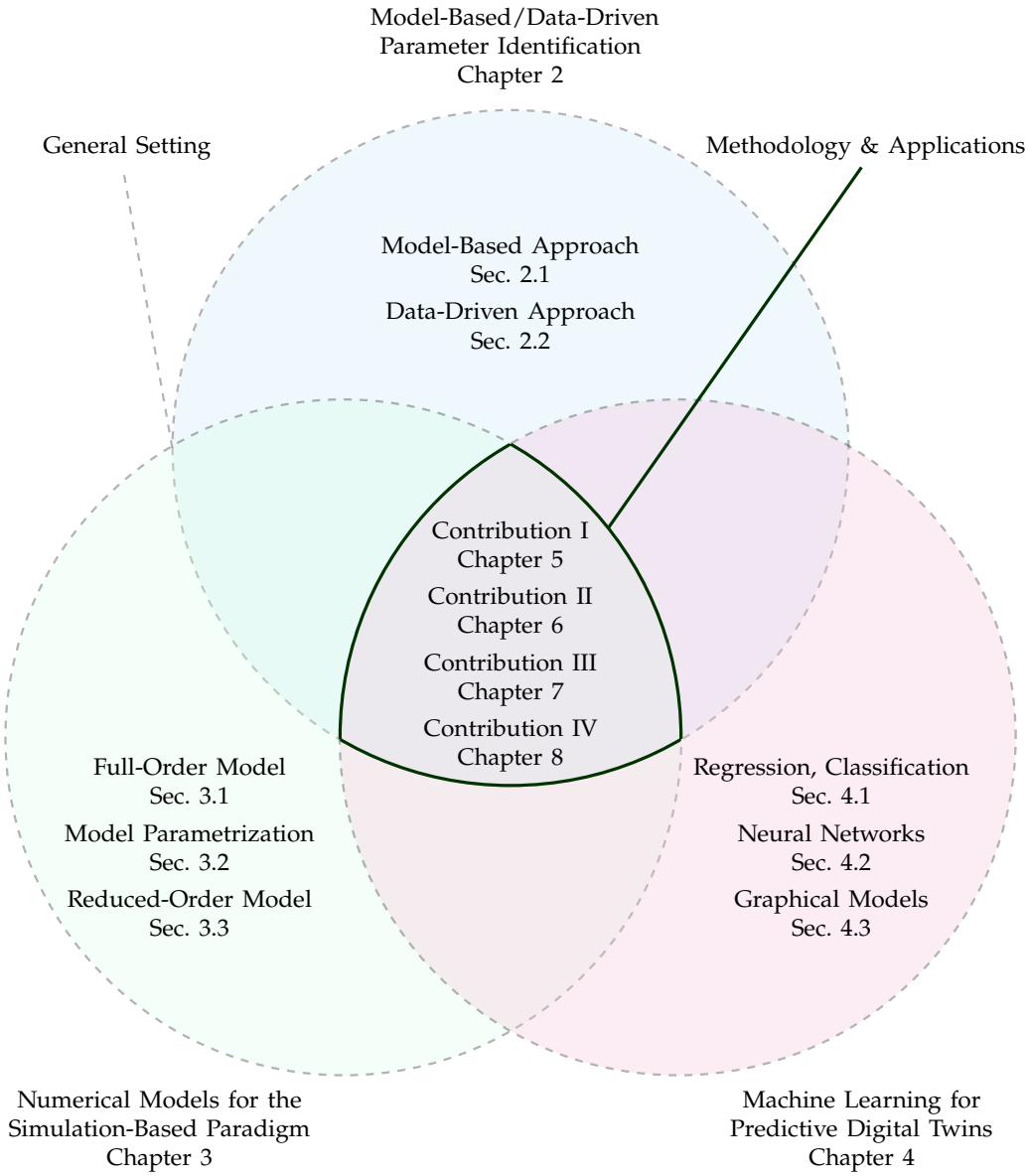


Figure 1.7: Graphical outline of the Thesis.

This page intentionally left blank.

2 MODEL-BASED AND DATA-DRIVEN APPROACHES TO PARAMETER IDENTIFICATION

We consider the identification of parameters from sensor measurements as a critical enabler for DTs of engineering structures. Model-based and data-driven approaches to SHM are two viable alternatives to address the inverse problem of identifying parameters from vibration recordings. In this Chapter, we exemplify the model-based vs. data-driven dichotomy by reviewing two well-established strategies for parameter identification that will appear several times throughout this Thesis: (*i*) model-based MCMC sampling and (*ii*) data-driven k-nearest neighbors regression. Comparing these two methods is useful for identifying the positive and negative aspects associated with each approach, as well as to provide a concrete basis for evaluating some of the objectives achieved in this Thesis.

The Chapter is structured as follows. The MCMC case is revisited in Sec. 2.1, with the description of parameter estimation from a Bayesian perspective in Sec. 2.1.1, followed by the introduction of the Metropolis and Metropolis-Hastings samplers in Sec. 2.1.2. The data-driven case of the k-nearest neighbors algorithm is detailed in Sec. 2.2, with a particular focus on the regression scenario.

Part of the research presented in this Chapter has been published as:

[164] *Torzoni et al., Structural health monitoring of civil structures: A diagnostic framework powered by deep metric learning,* Computers & Structures, *Vol. 271, 2022, p. 106858.*

[165] *Torzoni et al., A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks,* Mechanical Systems and Signal Processing, *Vol. 197, 2023, p. 110376.*

[166] *Torzoni et al., Enhancing Bayesian model updating in structural health monitoring via learnable mapping, 2024.* Under review.

[167] *Torzoni et al., 5 Damage identification using physics-based datasets: From convolutional to metric-informed damage-sensitive feature extractors,* Data-Centric Structural Health Monitoring: Mechanical, Aerospace and Complex Infrastructure Systems, De Gruyter, *2023, pp. 101–124.*

[168] *Torzoni et al., A Deep Neural Network, Multi-fidelity Surrogate Model Approach for Bayesian Model Updating in SHM,* European Workshop on Structural Health Monitoring, Springer International Publishing, *2023, pp. 1076–1086.*

[169] *Torzoni et al., Health Monitoring of Civil Structures: A MCMC Approach Based on a Multi-Fidelity Deep Neural Network Surrogate,* Computer Sciences & Mathematics Forum, *Vol. 2, No. 1, 2022, p. 16.*

2.1 A MODEL-BASED APPROACH: MCMC SAMPLING

2.1.1 PARAMETER ESTIMATION IN A BAYESIAN FRAMEWORK

When parameters are employed to model physical phenomena, empirical observations can be used to learn about the probability of parameters values, and such a dependence can be treated using *conditional probabilities*. Let us consider a parameter $\theta \in \mathbb{R}$ modeled as the random variable Θ, such that $\Theta \sim p(\theta)$, where $p(\theta)$ denotes the PDF encoding the probability that $\Theta = \theta$, and a batch of $N_{\text{obs}} \in \mathbb{N}$ observational data features $\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}} = [\mathbf{h}^{\text{EXP}}_1, \dots, \mathbf{h}^{\text{EXP}}_{N_{\text{obs}}}] \in \mathbb{R}^{N_{\text{obs}} \times N_h}$ that are independent realizations of the random variable $\mathbf{H}^{\text{EXP}} \sim p(\mathbf{h}^{\text{EXP}})$.

Each observation $\mathbf{h}^{\mathrm{EXP}}_{\mathscr{k}}$, $\mathscr{k} = 1, \ldots, N_{\mathrm{obs}}$, is the outcome of a noisy measurement experiment, described through a feature representation of size $N_h \in \mathbb{N}$. The batch size N_{obs} is assumed small enough to assume time-invariant operational, environmental, and damage conditions. According to Bayes' rule, the *posterior* PDF of θ conditioned on the batch of observational data features $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$, is given by:

$$p(\theta|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}) = \frac{p(\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta)p(\theta)}{\int p(\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta)p(\theta)\,\mathrm{d}\theta}. \tag{2.1}$$

For the SHM setting considered throughout this Thesis, the parameters of interest describe potential damage scenarios and operational conditions the structure might experience. Specific parametrization choices are discussed further in Sec. 3.2, with reference to the numerical models underlying the adopted simulation-based SHM paradigm. The observational data instead correspond to multivariate time series describing vibration recordings or, more conveniently, informative data features extracted from them. For instance, a feature extraction algorithm tailored to our SHM setting is proposed in Sec. 5.1, relying on deep metric learning.

The conditional PDF in Eq. (2.1) enables us to encode our posterior knowledge for θ given that we have observed $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$. In this context, the *prior* PDF $p(\theta)$ characterizes our initial belief on the values that the random variable Θ can take. This prior knowledge is embedded in the structure and regularity of $p(\theta)$ and may be based on heuristics, such as expert opinions, or derived from historical data and in situ inspections. In the cases where data are acquired *sequentially*, the posterior knowledge from the previous time becomes the prior for the current time. The term $p(\mathbf{H}^{\mathrm{EXP}} = \mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta)$ represents the *likelihood function*, which serves as the mechanism informing the posterior distribution about the observations. It describes the probability of observing the events $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$ given θ, through a suitable forward model linking θ and $\mathbf{h}$, as described in the following. Finally, the PDF $p(\mathbf{H}^{\mathrm{EXP}} = \mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}})$ in the denominator is termed *evidence* and serves as a normalizing constant, ensuring that the posterior PDF integrates to 1. Its computation is often avoided as it is very challenging, thus requiring alternative techniques for approximating the posterior PDF, as described below.

All Bayesian inverse problems require the same ingredients: (i) a forward model $\mathcal{M} : \mathbb{R} \mapsto \mathbb{R}^{N_h}$, $\theta \mapsto \mathbf{h}$, (ii) a set of input parameters θ to be inferred, and (iii) a set of experimental data $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$. The forward model $\mathcal{M}$ is a mathematical representation of the system under consideration, which can be statistical,

data-driven, mechanistic, or a combination of these. Often, this involves solving a differential problem using a FE method to simulate the dynamic response of a large-scale structure. For the SHM setting we consider in this Thesis, the forward models used to address Bayesian inverse problems are computationally efficient data-driven surrogate models: these models can target observable quantities, as shown by the MF-DNN-SMs developed in Chapter 6, or a set of informative features extracted from them, as proposed in Chapter 7. The model output $\widehat{\mathbf{h}} = \mathcal{M}(\theta)$ is an approximation of the real-world observation $\mathbf{h}^{\text{EXP}}$, interpreted through the following *additive observation model*:

$$\mathbf{H}^{\text{EXP}} = \mathcal{M}(\Theta) + \mathbf{H}_{\varepsilon}, \tag{2.2}$$

where $\mathbf{H}_{\varepsilon} \in \mathbb{R}^{N_h}$ is a *discrepancy term* accounting for modeling inaccuracy and measurement errors. In this Thesis, the random variable $\mathbf{H}_{\varepsilon} \sim p(\mathbf{h}_{\varepsilon})$ is modeled as *additive and mutually independent* from Θ. Moreover, it is also assumed to have zero mean, even if, more generally, it could also include a bias term.

In the context of Bayesian model inversion, the goal is to determine the optimal values of the parameter θ that enable fitting model predictions to the observations. Thanks to the assumption of mutual independence between Θ and $\mathbf{H}_{\varepsilon}$, $\mathbf{H}^{\text{EXP}}_{1,\dots,N_{\text{obs}}} - \mathcal{M}(\Theta)|\Theta = \theta$ is distributed like $\mathbf{H}_{\varepsilon}$, which can be expressed as:

$$p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta) = p_{\mathbf{H}_{\varepsilon}}(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}} - \mathcal{M}(\theta)), \tag{2.3}$$

where $p_{\mathbf{H}_{\varepsilon}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}$ denotes the PDF of the discrepancy term. Therefore, the specification of the likelihood function depends on the assumptions made regarding the distribution of $\mathbf{H}_{\varepsilon}$.

The simplest probabilistic model that can be used to describe experimental uncertainties is the *Gaussian* one. By assuming that the deviations due to modeling inaccuracies and measurement errors follow a Gaussian distribution $\mathbf{H}_{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\varepsilon})$, with zero mean and covariance matrix $\mathbf{\Sigma}_{\varepsilon} \in \mathbb{R}^{N_h \times N_h}$, the corresponding likelihood function becomes Gaussian too, and reads:

$$p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta) = \prod_{\mathscr{k}=1}^{N_{\text{obs}}} c^{-1}\text{EXP}\left(-\frac{(\mathbf{h}^{\text{EXP}}_{\mathscr{k}} - \mathcal{M}(\theta))^{\top}\mathbf{\Sigma}_{\varepsilon}^{-1}(\mathbf{h}^{\text{EXP}}_{\mathscr{k}} - \mathcal{M}(\theta))}{2}\right), \tag{2.4}$$

where $c = (\sqrt{2\pi})^{N_h}\sqrt{|\mathbf{\Sigma}_{\varepsilon}|}$ is a normalization constant.

In practical cases, the posterior distribution can not be evaluated analytically, and classical numerical integration methods also become infeasible, because the evidence term involves a

170. Beck *et al.*, *Updating Models and Their Uncertainties. I: Bayesian Statistical Framework.* 1998

high-dimensional integral that is hard to compute [170]. Alternative techniques for approximating the posterior PDF include *Laplace approximation-based methods* [171,172], *variational inference methods* [173,174], MCMC methods [175,176,177,178,179,180,181], and *filtering* methods [182,183].

Laplace approximation provides an asymptotic expansion of the posterior PDF under the condition that the system is either *globally or locally identifiable** [184]. The posterior PDF is approximated as a multivariate Gaussian distribution, centered either at the *maximum likelihood estimate* or the *maximum a-posteriori estimate*†, with the posterior covariance approximated by the inverse Hessian matrix of the negative log-likelihood evaluated at the posterior mean.

Variational inference provides an approximated solution by minimizing the *Kullback–Leibler divergence* [70] between a proposed PDF and the target PDF. The *mean field theory* [185] is commonly employed to factorize the proposed PDF and independently derive the posterior PDFs of each individual unknown. In combination with the *expectation maximization* algorithm, the variables and parameters in the variational inference model can be computed iteratively [186].

For the *unidentifiable cases*‡, if the probabilistic expression proportional to the posterior PDF in Eq. (2.1), namely:

$$p(\theta|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}) \propto p(\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta)p(\theta), \tag{2.5}$$

is available, an MCMC simulation enables a full characterization of the posterior PDF. MCMC methods directly establish the posterior PDF by generating samples through a *Markov chain*. However, basic MCMC techniques are extraordinarily time-consuming due to slow convergence and the repeated evaluation of computationally intensive forward models, thereby preventing real-time applications. More advanced samplers have been proposed to enhance computational efficiency and improve convergence rates. These include the *Metropolis-Hastings* algorithm, *Gibbs sampling*, *transitional* MCMC, *delayed rejection* technique, *hybrid Monte Carlo* algorithms, and even more recent extensions [187,188,189,190,191,192].

When data assimilation is performed sequentially, *filtering techniques* can be employed to treat model uncertainties and observation noise in a stochastic manner. The posterior PDF of *hidden state variables* is estimated given all the observations up to the current time. Due to its recursive updating scheme, filtering is typically preferred in place of batch-based MCMC sampling algorithms for applications where the data flow requires continuous assimilation as it gets collected or when nonlinear dynamic

171. Katafygiotis *et al.*, *A probabilistic approach to structural model updating*. 1998

172. Wang *et al.*, *Laplace approximation in sparse Bayesian learning for structural damage detection*. 2020

173. Li *et al.*, *Operational modal identification using variational Bayes*. 2017

174. Baldacchino *et al.*, *Variational Bayesian mixture of experts models and sensitivity analysis for nonlinear dynamical systems*. 2016

175. Lam *et al.*, *Markov chain Monte Carlo-based Bayesian method for structural model updating and damage detection*. 2018

176. Lye *et al.*, *An efficient and robust sampler for Bayesian inference: Transitional Ensemble Markov Chain Monte Carlo*. 2022

177. Betz *et al.*, *Transitional Markov Chain Monte Carlo: Observations and Improvements*. 2016

178. Mirzazadeh *et al.*, *Mechanical Characterization of Polysilicon MEMS: A Hybrid TMCMC/POD-Kriging Approach*. 2018

179. Rocchetta *et al.*, *On-line Bayesian model updating for structural health monitoring*. 2018

180. Meeds *et al.*, *GPS-ABC: Gaussian Process Surrogate Approximate Bayesian Computation*. 2014

181. Giannakeas *et al.*, *Digital clone testing platform for the assessment of SHM systems under uncertainty*. 2022

182. Cristiani *et al.*, *Damage diagnosis and prognosis in composite double cantilever beam coupons by particle filtering and surrogate modelling*. 2021

183. Eftekhar Azam *et al.*, *A dual Kalman filter approach for state estimation via output-only acceleration measurements*. 2015

*A system is considered identifiable when, with an infinite number of direct observations, the true hidden state of a deterministic system can be theoretically retrieved.

responses are considered, as seen in [193,194]. However, the basic *Kalman filter* formulation [195] is quite sensitive to numerical errors. An alternative formulation that is numerically more robust is the *square root Kalman filter* [196,197]. Nevertheless, a key drawback is that it can be applied only to linear observation and transition models. In cases requiring nonlinear models, the *unscented Kalman filter* and the *extended Kalman filter* allow for an approximation of the Kalman algorithm while describing hidden states using multivariate Gaussian random variables. Another option is given by *particle filters*, where hidden state variables are described by *particles* that propagate through the transition model, and their weights are updated according to the likelihood computed from the observation model. Particle filtering is well-suited for estimating PDFs that deviate from the multivariate Gaussian. Readers interested in these formulations can refer to [198,199,200] for further details.

2.1.2 METROPOLIS AND METROPOLIS-HASTINGS SAMPLERS

In this Section, we introduce two simple but intuitive MCMC algorithms, namely the *Metropolis* and *Metropolis-Hastings* samplers, which are employed in Chapter 6 and Chapter 7*. MCMC methods take their name from the *Markov property*, which states, roughly speaking, that:

> "Given the present, the future is independent of the past."

This property implies that if the current state of the system is known, future states can be predicted without having to consider past states. We employ a *time discretization*, and denote non-dimensional time steps by t. By modeling the target parameter θ as the random variable $\Theta_t \sim p(\theta_t)$, the transition to θ_{t+1} is governed by a *transition model* $p(\theta_{t+1}|\theta_t)$, which implies the following convenient equivalence (Markov property):

$$p(\theta_{t+1}|\theta_1, \theta_2, \dots, \theta_t) = p(\theta_{t+1}|\theta_t). \tag{2.6}$$

A *Markov chain* combines the Markov property and the chain rule of probability to define a joint distribution for random variables up to time $T > 1$, as the following stochastic process:

$$\begin{aligned} p(\theta_{1:T}) &= p(\theta_1)p(\theta_2|\theta_1)p(\theta_3|\theta_2)\dots p(\theta_T|\theta_{T-1}) = \\ &= p(\theta_1)\prod_{t=2}^{T} p(\theta_t|\theta_{t-1}). \end{aligned} \tag{2.7}$$

The idea behind MCMC is to construct a Markov chain with a *stationary* distribution corresponding to the posterior of θ, conditioned on a set of N_{obs} observational data features $\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}$.

†Instead of estimating the full posterior distribution using a Bayesian approach, the "maximum likelihood estimate" and the "maximum a-posteriori estimate" methods allow finding the optimal parameters that maximize the likelihood and the product of the likelihood and the prior, respectively.

70. Bishop, *Pattern Recognition and Machine Learning*. 2006

185. Parisi, *Statistical Field Theory*. 1998

186. Wang, Xiaoyou, *Probabilistic machine learning and Bayesian inference for vibration-based structural damage identification*. 2022

‡A deterministic system is considered unidentifiable when, even with an infinite number of direct observations, its true hidden state can not be retrieved, as multiple equally valid solutions exist.

187. Brooks *et al.*, *Handbook of Markov Chain Monte Carlo*. 2011

188. Haario *et al.*, *An Adaptive Metropolis Algorithm*. 2001

189. Geman *et al.*, *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*. 1984

190. Ching *et al.*, *Transitional Markov Chain Monte Carlo Method for Bayesian Model Updating, Model Class Selection, and Model Averaging*. 2007

191. Green *et al.*, *Delayed Rejection in Reversible Jump Metropolis-Hastings*. 2001

192. Hoffman *et al.*, *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*. 2014

193. Mariani *et al.*, *Impact induced composite delamination: state and parameter identification via joint and dual extended Kalman filters*. 2005

194. Bolzon *et al.*, *Parameter identification of a cohesive crack model by Kalman filter*. 2002

195. Kalman, *A New Approach to Linear Filtering and Prediction Problems*. 1960

196. Goulet, *Probabilistic Machine Learning for Civil Engineers*. 2020

197. Kaminski *et al.*, *Discrete square root filtering: A survey of current techniques*. 1971

198. Murphy, *Machine learning: A probabilistic perspective*. 2012

Conceptually, it involves randomly walking through the parameter space so that the fraction of steps spent at exploring each part of the support is proportional to the posterior PDF.

To ensure that the Markov chain serves as the stationary target distribution, it is necessary to guarantee the existence and uniqueness of an equilibrium solution. This requirement implies the need for the generated Markov chain to be *ergodic*, meaning *irreducibility*, *positive recurrence*, and *aperiodicity*†. If an MCMC algorithm produces such a chain, it converges to a unique stationary distribution. However, ensuring these conditions is not straightforward. Differently, *reversibility* is a sufficient but not necessary condition for an irreducible and aperiodic Markov chain to asymptotically tend to the stationary target distribution as $t \to \infty$. Reversibility can be established by ensuring that the solution satisfies the following relationship:

$$p(\theta_{t+1}|\theta_t)p(\theta_t|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}) = p(\theta_t|\theta_{t+1})p(\theta_{t+1}|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}), \tag{2.8}$$

which is the general *detailed balance condition*, suitably modified to have the target parameter θ conditioned on the observational data $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$. Eq. (2.8) implies that for a process at equilibrium, the number of times the process enters a state must be equal to the times it exits the same state in a fixed step interval.

Finding transition densities $p(\theta_{t+1}|\theta_t)$ and $p(\theta_t|\theta_{t+1})$ that satisfy Eq. (2.8) is not an easy task. The Metropolis and the Metropolis-Hastings algorithms address this challenge by decomposing the transition models into a user-defined *proposal density* $\widetilde{p} : \mathbb{R} \mapsto \mathbb{R}$ and an *acceptance ratio* $\alpha : (\mathbb{R}, \mathbb{R}) \mapsto \mathbb{R}$, resulting in the following relationship:

$$\begin{aligned}\widetilde{p}(\theta_{t+1}|\theta_t)\alpha(\theta_{t+1}|\theta_t)p(\theta_t|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}) = \\ \widetilde{p}(\theta_t|\theta_{t+1})\alpha(\theta_t|\theta_{t+1})p(\theta_{t+1}|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}),\end{aligned} \tag{2.9}$$

where, to satisfy the reversibility condition, $\alpha(\theta_t|\theta_{t+1}) = 1$.

A solution to Eq. (2.9) is given by:

$$\alpha(\theta_{t+1}|\theta_t) = \min\left(\frac{\widetilde{p}(\theta_t|\theta_{t+1})p(\theta_{t+1}|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}})}{\widetilde{p}(\theta_{t+1}|\theta_t)p(\theta_t|\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}})}, 1\right). \tag{2.10}$$

Recalling Bayes' equation, we find the following equivalence for the evidence term:

$$\begin{aligned}\int p(\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta_{t+1})p(\theta_{t+1})\,\mathrm{d}\theta_{t+1} = \\ \int p(\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\theta_t)p(\theta_t)\,\mathrm{d}\theta_t,\end{aligned} \tag{2.11}$$

199. Asch *et al.*, *Data assimilation: methods, algorithms, and applications*. 2016

200. Särkkä *et al.*, *Bayesian Filtering and Smoothing*. 2013

*While more advanced and effective samplers exist in the literature, this Thesis exclusively considers standard Metropolis and Metropolis-Hastings samplers. This decision is motivated by the high flexibility of these samplers and the chosen approach of enhancing model-based methods by adopting a data-driven perspective, rather than seeking sophisticated sampling techniques.

†In a discrete state space, a Markov chain is considered "irreducible" if all states communicate with each other. A state is labeled "recurrent" if it has a finite hitting time. Furthermore, the term "positive recurrent" is used for states with a finite average time between two visits. An irreducible Markov chain is also termed "aperiodic" if the return period to any given state is uncertain, with the greatest common divisor between the return times of two visits being equal to 1.

thus, by expanding $p(\theta_{t+1}|\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$ and $p(\theta_t|\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$ in Eq. (2.10) according to Eq. (2.1), we obtain:

$$\alpha(\theta_{t+1}|\theta_t) = \min\left(\frac{\widetilde{p}(\theta_t|\theta_{t+1})p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta_{t+1})p(\theta_{t+1})}{\widetilde{p}(\theta_{t+1}|\theta_t)p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta_t)p(\theta_t)}, 1\right), \tag{2.12}$$

providing the acceptance ratio for the Metropolis-Hastings algorithm. If a *symmetric proposal* is chosen, i.e., $\widetilde{p}(\theta_t|\theta_{t+1}) = \widetilde{p}(\theta_{t+1}|\theta_t)$, Eq. (2.12) simplifies to:

$$\alpha(\theta_{t+1}|\theta_t) = \min\left(\frac{p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta_{t+1})p(\theta_{t+1})}{p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta_t)p(\theta_t)}, 1\right), \tag{2.13}$$

giving the acceptance ratio for the Metropolis algorithm, which only depends on the ratio between the unnormalized posterior PDF evaluated at θ_{t+1} and θ_{t+1}.

Adopting the Metropolis sampler [201][‡], MCMC updates the prior PDF $p(\theta)$ to yield an unnormalized posterior PDF by iteratively generating a chain of samples $\{\theta_1, \dots, \theta_{L_{\text{chain}}}\}$ of length $L_{\text{chain}} \in \mathbb{N}$, drawn according to the proposal distribution $\widetilde{p}(\theta_{t+1}|\theta_t)$. The proposal must have a non-zero probability to transit from the current state to any state supported by the target distribution and must be symmetric[§].

201. Metropolis *et al.*, *Equation of State Calculations by Fast Computing Machines*. 1953

[‡]The Metropolis algorithm, developed during the Second World War while working on the Manhattan Project at Los Alamos, New Mexico, has become a foundational method in the field of Monte Carlo simulations.

[§]The Gaussian distribution is a commonly used general-purpose symmetric proposal distribution, where the mean is determined by the current position, and the likelihood of moving within a region around the current position is controlled by the covariance matrix.

The Metropolis algorithm requires defining an *initial state* θ_1 and a *target distribution* $\widehat{p}(\theta) = p(\mathbf{h}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\theta)p(\theta)$ corresponding to the unnormalized posterior PDF in Eq. (2.5) from which we want to sample. At each step, given a current position in the parameter space, the algorithm recursively employs the proposal distribution $\widetilde{p}(\theta_{t+1}|\theta_t)$ to propose a move to a new position $\overline{\theta}_{t+1} \in \mathbb{R}$. Subsequently, the algorithm decides whether to accept or reject each sample with probability $\alpha(\overline{\theta}_{t+1}|\theta_t)$. If the target distribution $\widehat{p}(\theta)$ evaluated at the proposed location is greater than or equal to the current one, i.e., $\widehat{p}(\overline{\theta}_{t+1}) \geq \widehat{p}(\theta_t)$, we accept the proposed location, and $\theta_{t+1} = \overline{\theta}_{t+1}$. If the proposed location has a target value lower than the current one, we accept moving to the proposed location with a probability equal to the acceptance ratio $\alpha(\overline{\theta}_{t+1}|\theta_t)$. If the proposed location is rejected, we stay at the current location, and $\theta_{t+1} = \theta_t$. After a certain amount of iterations L_{chain}, to be chosen as detailed later, the fraction of steps spent exploring each part of the support is proportional to the density of the target distribution $\widehat{p}(\theta)$. The Metropolis sampling algorithm is formalized in Algorithm 2.1.

The Metropolis-Hastings algorithm [202] is identical to the Metropolis algorithm, with the key distinction that it allows for *non-symmetric* transition probabilities $\widetilde{p}(\theta_t|\theta_{t+1}) \neq \widetilde{p}(\theta_{t+1}|\theta_t)$.

202. Hastings, *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*. 1970

Algorithm 2.1 Metropolis sampling

Input: prior distribution $p(\theta)$
target distribution $\widehat{p}(\theta)$
proposal distribution $\widetilde{p}(\theta_{t+1}|\theta_t)$
chain length L_{chain}
initial state θ_1

for $t = 1 : L_{\text{chain}}$ **do**
  draw trial sample $\overline{\theta}_{t+1}$ according with $\widetilde{p}(\theta_{t+1}|\theta_t)$
  compute acceptance ratio $\alpha(\overline{\theta}_{t+1}|\theta_t) = \min(\frac{\widehat{p}(\overline{\theta}_{t+1})}{\widehat{p}(\theta_t)}, 1)$
  if $\overline{\theta}_{t+1}$ accepted with probability $\alpha(\overline{\theta}_{t+1}|\theta_t)$ **then**
    $\theta_{t+1} = \overline{\theta}_{t+1}$
  else
    $\theta_{t+1} = \theta_t$
return chain of samples $\{\theta_1, \ldots, \theta_{L_{\text{chain}}}\}$

The main difference in the Metropolis-Hastings is that, when the proposed location has a target value that is lower than the current location, the proposed location is accepted with a probability equal to $\frac{\widehat{p}(\theta_{t+1})}{\widehat{p}(\theta_t)} \cdot \frac{\widetilde{p}(\theta_t|\theta_{t+1})}{\widetilde{p}(\theta_{t+1}|\theta_t)}$. This situation arises, for example, when using a truncated Gaussian PDF as proposal.

Convergence check. The random walk can be significantly influenced by its initialization for numerous iterations. Depending on the initial state and transition PDF, the sampling procedure may remain trapped in a specific part of the support, causing the sample's density to be unrepresentative of the posterior PDF. Moreover, challenges may arise in the presence of multi-modal target distributions, as the random walk may take a considerable amount of time to transition from one mode to another. In practice, it is common to discard the initial transient phase, referred to as the *burn-in phase* of the chain, and then perform a *convergence check* to assess whether the retained MCMC samples are realizations from a stationary distribution.

The convergence of the chain to the stationary distribution describing the posterior PDF can only be guaranteed asymptotically, as $L_{\text{chain}} \to \infty$. In practice, after a finite number of draws, the resulting distribution lies in between the starting and target distributions [203]. Diagnostics must be applied to check whether the MCMC samples belong to a stationary distribution. The number of draws necessary to meet the convergence criterium is a-priori unknown, and the speed of convergence depends on the acceptance rate of the algorithm. Since it is generally not possible to monitor the convergence of an MCMC simulation from a single chain [204], the *estimated potential scale*

203. Gelman *et al.*, *Inference from Iterative Simulation Using Multiple Sequences*. 1992

204. Gelman *et al.*, *A single series from the Gibbs sampler provides a false sense of security*. 1992

reduction (EPSR) [203] diagnostic exploits multiple randomly initialized chains from parallel runs to monitor on-the-fly the convergence to a stationary distribution. Only when all the chains converge to the same stationary distribution, the convergence criterium is satisfied. This diagnostics allows to check the quality of the estimates and to stop MCMC as soon as possible.

The EPSR metric $\widehat{R} \in \mathbb{R}$ assesses the convergence of a multivariate chain by comparing the *estimate of the between-chain variance* of samples $\widehat{V}_R \in \mathbb{R}$ to the *average within-chain variance* of samples $W_R \in \mathbb{R}$. Since W_R tends to underestimate the true variance of samples and $\widehat{V}_R$ is prone to overestimate it, when all the chains approach the same equilibrium distribution, $\widehat{R} = \sqrt{\frac{\widehat{V}_R}{W_R}} \to 1$. Conversely, if the chains have not yet converged to a common distribution, $\widehat{R} > 1$. According to [203], convergence can deemed to be met when $\widehat{R} < R_\varepsilon$, with $R_\varepsilon = 1.1$ being a recommended tolerance value.

Upon meeting the convergence criterion and achieving a stable Markov chain, the chain is subsequently thinned to reduce the within-chain autocorrelation of samples. The posterior PDF is ultimately approximated using histograms, and the expected values and covariance of parameters are approximated by means of their empirical counterparts. The maximum a-posteriori estimate, obtained by maximizing the posterior distribution, is also used often as a point estimate of the parameter value, although this implies solving an additional optimization problem.

Adaptive MCMC. For the Metropolis or Metropolis-Hastings algorithm, the bottleneck in achieving rapid convergence typically lies in defining the proposal distribution $\widetilde{p}(\theta_{t+1}|\theta_t)$, which should closely match the underlying target distribution. An inappropriate proposal can result in an inefficient implementation, requiring a large number of samples to meet the convergence criterion. For a Gaussian proposal distribution, the key factor for obtaining an accurate representation of the posterior PDF is the variance of the proposal. If the variance is too large, the new candidates mostly miss the target portion of the support and are rarely accepted. If the variance is too small, the new candidates are mostly accepted but only from a small neighborhood of the previous point, leading to slow movement.

A good starting point for selecting the proposal PDF is to use a Gaussian PDF centered on the current state, with the covariance matrix approximated as the posterior covariance using the Laplace approximation calculated for the maximum a-posteriori value. However, this strategy may not always lead to efficient sampling, especially when some parameters are

poorly identified by the available data. A viable alternative is *adaptive* MCMC, which allows tuning the proposal on-the-run as the sampling proceeds, incorporating information from previously sampled points. In the adaptive Metropolis sampling algorithm [188,205,206], the proposal density is a Gaussian PDF centered at the current location, with covariance matrix initialized as an arbitrary strictly positive definite matrix with entries sufficiently small to set the sampler in motion. The covariance is then tuned iteratively as the sampling progresses, using the empirical covariance matrix of the previously sampled points. This adaptation does not require an update after every step, thus allowing a limited computational cost. In the long rum, the adaptive Metropolis reverts to non-adaptive sampling, as new sampled points have less impact on the proposal.

188. Haario *et al.*, *An Adaptive Metropolis Algorithm*. 2001

205. Andrieu *et al.*, *A tutorial on adaptive MCMC*. 2008

206. Haario *et al.*, *Adaptive proposal distribution for random walk Metropolis algorithm*. 1999

2.2 A DATA-DRIVEN APPROACH: K-NEAREST NEIGHBORS

In contrast to the model-based MCMC algorithms discussed in Sec. 2.1.2, this Section delves into a data-driven approach, which is the *k-nearest neighbors* (k-NN) algorithm [207,208,209] employed in Chapter 5 and Chapter 6. It is a non-parametric, memory-based method that operates swiftly during both the learning and the prediction stages, especially when dealing with limited data dimensionality. By relying on the concept of *similarity* between close data points, this algorithm is suitable for both classification and regression tasks. A key distinction between the k-NN and MCMC algorithms lies in their approach to the problem, highlighting the model-based and data-driven dichotomy. Moreover, while MCMC sampling naturally incorporates uncertainties and treats the system as probabilistic, the k-NN algorithm adopts a deterministic perspective, neglecting the uncertainties associated with the problem.

207. Fix *et al.*, *Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties*. 1989

208. Altman, *An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression*. 1992

209. Cover *et al.*, *Nearest neighbor pattern classification*. 1967

In both classification and regression settings, the k-NN inputs typically consist of query data features. The target output is computed by finding the $N_k \in \mathbb{N}$ closest training data points to the query. These are identified by computing mutual distances between the training data points and the query in the feature space, typically using the *Euclidean* or the *Mahalanobis* distance metric. In *k-NN classification*, the output is the most frequent class membership among the identified N_k nearest neighbors. If $N_k = 1$, then the query is simply assigned to the class of its single nearest neighbor. In *k-NN regression*, the output is computed as the average of the values associated with the N_k nearest neighbors. If $N_k = 1$, then the query is simply assigned

to the value of its single nearest neighbor.

For both classification and regression, a common *weighting scheme* consists of giving each neighbor a weight, so that the nearer neighbors contribute more to the prediction than the more distant ones. This also helps in mitigating potential issues related to skewed distributions of data labels, for which a more frequent class may dominate the prediction.

For high-dimensional data, the algorithm is typically applied to data features to avoid effects related to the *curse of dimensionality**. *Dimensionality reduction* in the form of feature extraction is commonly employed to produce a low-dimensional embedding of the data before applying the k-NN algorithm. If data features represent different quantities or show different scales, normalizing the feature space can significantly improve the accuracy of the k-NN algorithm.

*In the context of the k-nearest neighbors algorithm, the curse of dimensionality involves the query data point being nearly equidistant from all the training data points, making the computed distances uninformative.

During training, the k-NN model $\mathcal{M}_{\text{k-NN}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}$, $\mathbf{h} \mapsto \theta$, stores $N_{\text{train}} \in \mathbb{N}$ data points, or training *instances*, collected as:

$$\mathcal{D} = \{(\mathbf{h}_i, \theta_i)\}_{i=1}^{N_{\text{train}}}, \tag{2.14}$$

where $\mathbf{h}_i$ are the training data features, and θ_i the corresponding labels, with $i = 1, \ldots, N_{\text{train}}$. At prediction time, the algorithm computes a distance $\mathcal{E}_{\mathbf{h}} : (\mathbb{R}^{N_h}, \mathbb{R}^{N_h}) \mapsto \mathbb{R}$, $(\mathbf{h}^{\text{EXP}}, \mathbf{h}_i) \mapsto \mathbb{R}$, among the query data point $\mathbf{h}^{\text{EXP}}$ and $\mathbf{h}_i$, for $i = 1, \ldots, N_{\text{train}}$. The labeled data points are then ordered by increasing distance to compute an estimate $\widehat{\theta}$ as a weighted average over the N_k nearest neighbors, according with:

$$\widehat{\theta} = \mathcal{M}_{\text{k-NN}}(\mathbf{h}^{\text{EXP}}) = \frac{\sum_{k=1}^{N_k} \chi_k \mathcal{E}_{\mathbf{h}}(\mathbf{h}^{\text{EXP}}, \mathbf{h}_k)\theta_k}{\sum_{k=1}^{N_k} \chi_k \mathcal{E}_{\mathbf{h}}(\mathbf{h}^{\text{EXP}}, \mathbf{h}_k)}. \tag{2.15}$$

In Eq. (2.15), $\chi_k \in \mathbb{R}$, with $k = 1, \ldots, N_k$, is the distance weight associated with the k-th closest neighbor of $\mathbf{h}^{\text{EXP}}$. The weighting rule can be uniform, i.e., $\chi_k = 1 \; \forall k$, or based on the inverse distance, i.e., $\chi_k = \frac{1}{\mathcal{E}_{\mathbf{h}}(\mathbf{h}^{\text{EXP}}, \mathbf{h}_k)}$, where $\mathbf{h}_k \in \mathcal{D}$ is the k-th closest neighbor of $\mathbf{h}^{\text{EXP}}$.

Cross-validation strategy. The selection of the number of neighbors N_k in the k-NN algorithm depends on the specific characteristics of the data. An optimal N_k value, along with the relative weighting rule, can be determined through a *cross-validation strategy*, which aims to identify the *model of optimal complexity*. This strategy works as follows [196]. The training data are randomly divided into $N_{\text{folds}} \in \mathbb{N}$ distinct folds. The model is then trained and evaluated N_{folds} times, with a different fold chosen for evaluation each time, while the others are used

196. Goulet, *Probabilistic Machine Learning for Civil Engineers*. 2020

for training. For each possible combination of N_k and weighting rule, a total regression score is calculated by averaging over the N_{folds} validation folds. The model class that provides the maximum fold-averaged score is selected and retrained using the entire dataset.

Performance indicators. When employed for classification purposes, the k-NN performance can be assessed using the *classification accuracy* indicator:

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{2.16}$$

measuring the ratio of correct classifications to the total number of classifications, where $\text{TP} \in \mathbb{N}$ denotes true positives, $\text{TN} \in \mathbb{N}$ true negatives, $\text{FP} \in \mathbb{N}$ false positives, and $\text{FN} \in \mathbb{N}$ false negatives.

When employed for regression, the k-NN performance can be assessed by monitoring the *mean absolute error* (MAE) and the *root mean squared error* (RMSE) indicators, respectively:

$$\text{MAE} = \frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} |\theta_j - \widehat{\theta}_j|, \tag{2.17}$$

$$\text{RMSE} = \sqrt{\frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} (\theta_j - \widehat{\theta}_j)^2}, \tag{2.18}$$

where $N_{\text{test}} \in \mathbb{N}$ is the number of testing instances. The MAE is a common metric used to assess the accuracy of a regression model, providing a straightforward interpretation of the prediction accuracy that is not very sensitive to outliers. Similarly, the RMSE is also a common metric to assess the accuracy of a regression model. It is particularly sensitive to outliers due to the squaring operation, making it a useful metric to use in conjunction with the MAE.

2.3 THESIS PERSPECTIVE

The comparison between the k-NN workflow presented above and the MCMC sampling methods described in Sec. 2.1.2 highlights the model-based and data-driven dichotomy. The k-NN algorithm encompasses both offline and online phases, as encapsulated in Eq. (2.14) and in Eq. (2.15), respectively. By automatically learning a relationship between data features and quantities of interest, data-driven approaches, such as the k-NN algorithm, typically excel in handling large amounts of raw noisy data compared to model-based approaches. Moreover, since the

statistical relationship is learned during training, data-driven approaches usually allow faster predictions. This stands in contrast to the MCMC case, where there is no need to generate a training dataset*, and prior knowledge is encoded within the forward model used to assess the likelihood function in Eq. (2.3).

*This statement refers to the general naive case where model-based approaches to inverse problems are not improved in efficiency using cheap-to-evaluate data-driven surrogate models.

Based on these considerations, we will leverage on both approaches in this Thesis, however at different stages. In particular, a data-driven approach relying on DL models for automatic feature extraction is proposed in Sec. 5.1. This strategy allows encoding the sensitivity to the sought parameters in a learnable metric space, enabling a manifold to capture the variability of the structural system. In Sec. 5.2, we then exploit the learned low-dimensional feature space to feed a data-driven k-NN regressor for real-time damage localization. This setup, relying on a DL feature extractor and a k-NN regressor, is also used in Sec. 6.3 in the context of the proposed MF-DNN surrogate modeling strategy. In this case, the combined use of DL algorithms and k-NN regression is employed for the preliminary identification of operational conditions. The identification outcome serves as an informative prior knowledge for the model-based MCMC sampling of SHM parameters. This informative prior not only keeps Bayesian inference within a plausible range but also serves to initialize the Markov chains in high probability regions, thereby helping the MCMC convergence. Here, the data-driven MF-DNN-SM is exploited to speed up the evaluation of the likelihood function while sampling the posterior distribution of SHM parameters. The *hybrid* data-driven/model-based approach proposed in Sec. 7.1 instead capitalize on the strengths of both approaches to enhance stochastic SHM methodologies. This strategy combines the probabilistic model-based MCMC framework with the automatic feature extraction capabilities enabled by data-driven DL models. Finally, an additional data-driven damage identification strategy is presented in Sec. 8.2 within the proposed framework for predictive DTs. In this case, the damage assessment is entirely addressed with DL models, employing DL classifiers for damage detection and localization, and DL regressors for damage quantification. While in the rest of this Thesis we approach the parameter identification problem from a regression perspective, this is the single instance in which we frame the damage localization problem as a classification task — a change of perspective that is motivated by the discretized setting characterizing the asset-twin PGM.

This page intentionally left blank.

3 NUMERICAL MODELS FOR THE SIMULATION-BASED PARADIGM

In this Chapter, we focus on physics-based computational models, providing the foundation for the development of damage identification strategies within the simulation-based paradigm of SHM. Physics-based models offer a knowledge platform where the effect of damage on the structural response can be systematically replicated. Throughout this Thesis, structural response data describing vibration recordings, such as displacements or accelerations shaped as *multivariate time series*, are simulated and collected in a structural dynamics setting to emulate the output of a monitoring system. Specifically, the reference FOM is established using a FE-based semi-discretized form of the elasto-dynamic problem.

Establishing a suitable *parametrization of the structural model*, vibration data are simulated and collected under various operational and damage conditions. Without loss of generality, we refer to a baseline condition representing the initial monitoring phase of an undamaged structure. Each variable within the parameteric input space is treated as a random variable, allowing to embed prior knowledge regarding structural behavior, operational conditions, and potential damage scenarios into the sampling of the parametric input space.

Many-query SHM applications can imply high computational costs. This can be due to the repeated evaluation of a computationally intensive forward model in model-based approaches, or the need to populate a sufficiently representative synthetic dataset in data-driven approaches.

To mitigate the risk of a blowing computational cost for large scale structures, a *projection-based model order reduction strategy for parametrized systems* is employed to construct an LF model. Specifically, the FOM of the monitored structure is replaced with a computationally efficient ROM using a proper orthogonal decomposition (POD)-Galerkin reduced basis method for parametrized FE models.

In distinguishing between FOM and ROM problems, we have referred to them as HF and LF models, respectively. This distinction does not imply that reduction techniques refer to

Part of the research presented in this Chapter has been published as:

[94] *Rosafalco et al., Online structural health monitoring by model order reduction and deep learning algorithms,* Computers & Structures, *Vol. 255, 2021, p. 106790.*

[106] *Torzoni et al., SHM under varying environmental conditions: An approach based on model order reduction and deep learning,* Computers & Structures, *Vol. 266, 2022, p. 106790.*

[100] *Rosafalco et al., A Self-adaptive Hybrid Model/data-Driven Approach to SHM Based on Model Order Reduction and Deep Learning,* Structural Health Monitoring Based on Data Science Techniques, Springer International Publishing, *2022, pp. 165–184.*

[210] *Parola et al., Convolutional Neural Networks for Structural Damage Localization on Digital Twins,* Deep Learning Theory and Applications, Springer International Publishing, *2023, pp. 78–97.*

[105] *Torzoni et al., A Combined Model-Order Reduction and Deep Learning Approach for Structural Health Monitoring under Varying Operational and Environmental Conditions,* Engineering Proceedings, *Vol. 2, 2020, p. 94.*

less accurate discretization or simplified physics. The ROMs considered in this Thesis are still physics-based, and they can be equipped with error estimates or error bounds. Indeed, the fundamental difference between FOMs and ROMs lies in the dimension of the subspace where the solution is sought. This aspect allows us to speed up structural analyses and data assimilation while keeping explicit the parametrization of the structural models to a wide range of structural states that may be encountered during operations.

The Chapter is structured as follows. We first introduce the reference FOM in Sec. 3.1. Next, we provide details on the parametrization of damage and operational conditions, along with the corresponding sampling strategy, in Sec. 3.2. Finally, we elaborate on the employed model order reduction procedure and the benefits this approach affords in Sec. 3.3.

3.1 FULL-ORDER MODEL

The structures under consideration, intended for health monitoring, are modeled as linear-elastic continua that are discretized in space using FEs. Their dynamic response to the applied loadings, assuming linearized kinematics, is characterized by the *semi-discretized form of the elasto-dynamic problem*, represented as the following second-order system of ordinary differential equations and related set of initial conditions:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{x}}(t) + \mathbf{C}(\boldsymbol{\mu})\dot{\mathbf{x}}(t) + \mathbf{K}(\boldsymbol{\mu})\mathbf{x}(t) = \mathbf{f}(t, \boldsymbol{\mu}), \quad t \in (0, T) \\ \mathbf{x}(0) = \mathbf{x}_0, \\ \dot{\mathbf{x}}(0) = \dot{\mathbf{x}}_0, \end{cases} \tag{3.1}$$

which serves as the FOM enabling the computation of solution trajectories for comparison with sensor recordings. In Eq. (3.1), $t \in \mathbb{R}$ denotes time and, for the problem setting we consider, the time interval $(0, T)$, for a finite time horizon $T > 0$, is short enough for the operational, environmental, and damage conditions to be considered time-invariant, yet long enough to not compromise the identification of the structural behavior. The vectors $\mathbf{x}(t), \dot{\mathbf{x}}(t), \ddot{\mathbf{x}}(t) \in \mathbb{R}^{N_{\text{FE}}}$ denote the *nodal displacements, velocities and accelerations*, respectively, where N_{FE} is the number of DOFs. The matrix $\mathbf{M} \in \mathbb{R}^{N_{\text{FE}} \times N_{\text{FE}}}$ is the *mass matrix*. It is worth noting that $\mathbf{M}$ is independent of $\boldsymbol{\mu}$ since the mass properties of the structure are considered unaffected by the employed damage description or operational conditions. The matrix $\mathbf{C}(\boldsymbol{\mu}) \in \mathbb{R}^{N_{\text{FE}} \times N_{\text{FE}}}$ is the *damping matrix*, which is assembled according to the Rayleigh's model. The matrix $\mathbf{K}(\boldsymbol{\mu}) \in \mathbb{R}^{N_{\text{FE}} \times N_{\text{FE}}}$ is the *stiffness matrix*. The vector $\mathbf{f}(t, \boldsymbol{\mu}) \in \mathbb{R}^{N_{\text{FE}}}$ represents nodal

forces induced by the external loadings. The initial conditions at $t = 0$, in terms of nodal displacements and velocities, are denoted as $\mathbf{x}_0$ and $\dot{\mathbf{x}}_0$, respectively. The vector $\boldsymbol{\mu} \in \mathbb{R}^{N_{\text{par}}}$ includes $N_{\text{par}} \in \mathbb{N}$ parameters, representing the variability of the monitored system in terms of structural health and operational conditions.

Adopting a uniform partition of the time interval $(0, T)$ into $L \in \mathbb{N}$ time steps, the solution to Problem (3.1) is advanced in time using an *implicit Newmark integration scheme* [211]. This yields $\mathbf{x}_l$, $\dot{\mathbf{x}}_l$ and $\ddot{\mathbf{x}}_l$, for $l = 1, \ldots, L$, namely the vectors of nodal displacements, velocities and accelerations at time l. By choosing the *unconditionally stable constant average acceleration integration rule* [212], we can arbitrarily choose the integration step on the basis of the structural frequencies and the sensor sampling rate.

211. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. 2000

212. Newmark, *A Method of Computation for Structural Dynamics*. 1959

3.2 DAMAGE AND OPERATIONAL CONDITIONS

In the domain of civil structures, our focus lies on identifying damage patterns characterized by a small evolution rate. These damage patterns must be identified independently from confounding influences related to measurement noise, environmental variability, and operational variability. In this context, *operational variability* refers to any change in the loads acting on the monitored structure, such as variations in amplitude and frequency for harmonic loading components, changes in weight and speed for moving loads, and fluctuations in magnitude and epicentral distance for low-intensity seismic events. All these variations can be addressed through the parameter vector $\boldsymbol{\mu}$.

As discussed in Sec. 1.2.1, changes in the structural dynamic response due to damage are commonly modeled through a localized reduction of the effective stiffness. This damage description is valid as long as the underlying assumption, that the structure continues to behave as a linear time-invariant system within an observation window after the inception of damage, holds. Such an assumption is generally accepted for civil structures. However, it would be incorrect if the structural response is nonlinear, such as in structures experiencing large displacements or affected by fast-propagating damage patterns. Throughout this Thesis, local stiffness reduction is achieved by parametrizing the stiffness matrix through a set of parameters included in $\boldsymbol{\mu}$. The damage condition remains fixed during the time interval $(0, T)$, allowing for a *timescale separation* between damage evolution and health assessment. In particular, two distinct parametrizations are considered, depending on whether damage localization is approached from a classification or a regression perspective.

In the classification case, the stiffness matrix is parametrized

using two variables, $\ell \in \mathbb{N}$ and $\delta \in \mathbb{R}$, both included in the parameter vector $\boldsymbol{\mu}$. Here, $\ell \in \{0, \ldots, N_\ell\}$ represents the specific *damage scenario*, among a set of $N_\ell \in \mathbb{N}$ *predefined* damage locations (where $\ell = 0$ denotes the damage-free baseline), while δ is the associated *magnitude of the stiffness* reduction*. Adopting a regression perspective for damage localization, the stiffness matrix is instead parametrized using the *spatial coordinates* of the subdomain wherein the stiffness is reduced, denoted as $\mathbf{y} \in \mathbb{R}^3$. It is worth noting that the choice between a classification or regression strategy is independent of the simulation-based approach employed in this Thesis, allowing for the synthesis of labeled supervision through numerical simulations.

*This type of parametrization is referred to as "affine parametrization", allowing the expression of the underlying parametrized partial differential equation as a sum of parameter-dependent functions, multiplied by parameter-independent operators. In cases where a non-affine parametrization is chosen, an approximate affine expansion can be recovered using the empirical interpolation method [213].

Every time Problem (3.1) has to be solved, it is first required to draw a $\boldsymbol{\mu}$ sample. Basic sampling techniques are typically based on *pseudo-random number generation*†. Under the assumption of independent random variables, this family of strategies samples each random variable in the parametric input space from the corresponding marginal distribution. The associated risk is that the parametric space is not evenly filled, with the resulting samples exhibiting clusters and gaps. This drawback can be mitigated by increasing the number of samples. However, requiring a large number of samples makes random sampling inefficient when dealing with a computationally expensive FOM.

†Pseudo-random number generation produces a sequence of numbers, whose properties approximate those of random numbers over the interval $[0, 1]$. However, this can be readily applied to a random variable defined over a different support by transforming it into a random variable with probability density function defined over $[0, 1]$.

Stratified sampling is a family of sampling techniques consisting, first, in dividing the parametric space into subspaces containing the same number of samples, and then randomly sampling from each subspace. While a partition ensures in principle a good coverage of the parametric space, random sampling from each subspace guarantees sufficient randomness [214]. The good compromise between randomness and even coverage of the parameteric space depends on the stratification of the parameter space [215] and is particularly appreciable for a limited number of samples.

214. Hesthaven *et al.*, *Non-intrusive reduced order modeling of nonlinear problems using neural networks*. 2018

215. Saltelli *et al.*, *Global sensitivity analysis: the primer*. 2008

Throughout this Thesis, we employ the *Latin hypercube sampling rule*, which is an efficient stratified sampling technique [216]. Latin hypercube sampling subdivides each marginal PDF underlying the parameter space, so that each sub-interval is sampled exactly once. However, it is important to note that knowing the number of samples is required at design stage to subdivide the support of each marginal PDF‡.

216. McKay *et al.*, *A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code*. 1979

‡As an alternative to the employed Latin hypercube rule, "low-discrepancy sequences" techniques do not suffer from the drawback of requiring prior knowledge about the number of needed samples.

In this Thesis, the parametric input spaces are assumed to have a uniform marginal PDF for each parameter, although there are no restrictions in this regard. Other probability distributions can also be considered to incorporate any a-priori knowledge of the structure and relevant operational conditions. For instance,

this knowledge may arise from a former inspection of the structure to determine if a specific damage state is more likely to occur than others, or from a global sensitivity analysis to provide insights into the relationships between loading conditions and the probability of initiating a specific damage pattern.

3.3 REDUCED-ORDER MODELING

As the number of DOFs N_{FE} in the FOM increases, the computational cost of solving Problem (3.1) for any sampled $\boldsymbol{\mu}$ also grows, making the systematic generation of synthetic structural response data prohibitive. In this context, model order reduction techniques for parametrized systems, such as the reduced basis method, can be exploited to speed up the solution of an HF-FOM with a more affordable yet accurate approximation obtained through a projection-based ROM. Readers interested in formulations for projection-based ROMs can refer to [217,218,219] for further details. In this Thesis, we employ the *POD-Galerkin reduced basis method for parametrized FE models*, exploiting the so-called *method of snapshots* [220]. This strategy allows preserving the dependence of the solution from input parameters and time while controlling the level of introduced approximation.

217. Benner *et al.*, *A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems*. 2015

218. Quarteroni *et al.*, *Reduced Order Methods for Modeling and Computational Reduction*. 2014

219. Chinesta *et al.*, *Model Reduction Methods*. 2017

220. Sirovich, *Turbulence and the dynamics of coherent structures. I. Coherent structures*. 1987

The remainder of this Section is organized as follows. In Sec. 3.3.1, we first review the mathematical formulation of POD, also explaining how the decomposition is computed and providing insights into the physical interpretation of the extracted POD basis functions. The POD-Galerkin reduced basis method for parametrized elasto-dynamics problems is then detailed in Sec. 3.3.2. References for the employed equations describing the POD-based reduced basis method can be found in [124,125,221].

124. Kerschen *et al.*, *The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview*. 2005

125. Quarteroni *et al.*, *Reduced Basis Methods for Partial Differential Equations: An Introduction*. 2015

221. Golub *et al.*, *Matrix computations*. 2013

3.3.1 PROPER ORTHOGONAL DECOMPOSITION

POD is a matrix analysis technique designed to achieve a concise representation of data, serving two primary purposes: (i) dimensionality reduction by projecting high-dimensional data into a lower-dimensional space, and (ii) feature extraction by unveiling relevant structures within the data. The fundamental concept of POD involves the reduction of several interdependent variables to a much smaller set of *uncorrelated variables*, while maximizing the preserved *information content*. This is accomplished through an orthogonal transformation to the basis of the eigenvectors of the sample covariance matrix. The data are then projected onto the subspace defined by the eigenvectors corresponding to the largest eigenvalues. This transformation decorrelates signal components and maximizes the variance. The remarkable

feature of POD lies in its *optimality*, minimizing the average squared distance between the original signal and its reduced linear representation. Among all the possible linear subspaces of same dimension, POD provides the optimal approximating *linear manifold* in the configuration space represented by the data, yet it presents notable limitations when dealing with data situated on a *nonlinear manifold*.

Historically, POD was independently introduced by distinct authors in different fields and under different names, including *principal component analysis* in the statistical and dimensionality reduction literature [222,223], and *Karhunen-Loève decomposition* in the analysis of stochastic processes [224]. In [225,226,227], the equivalence of POD, principal component analysis and Karhunen-Loève decomposition is discussed, highlighting their connection with *singular value decomposition*.

POD has undergone extensive investigation in the context of structural dynamics [124,228,229,230,231,232] and structural analysis [233,234,235], providing clear geometric, energetic and physical interpretations of the POD basis functions. Moreover, POD benefits from efficient criteria for selecting the basis functions and quantifying the approximation errors. These POD basis functions, also known as *proper orthogonal modes* (POMs), are obtained exploiting the appealing *offline-online decoupling* of POD. Indeed, an investment of computational time is required during the offline phase to develop a low-dimensional model, which can then be rapidly evaluated in the online phase.

Given any matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_{N_C^{\mathbf{A}}}] \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_C^{\mathbf{A}}}$, collecting $N_C^{\mathbf{A}} \in \mathbb{N}$ zero-mean observations with dimension $N_R^{\mathbf{A}} \in \mathbb{N}$, or *snapshots*, POD employs a matrix factorization called singular value decomposition (SVD), which can be written as:

$$\mathbf{A} = \mathbf{P}\mathbf{\Sigma}\mathbf{Z}^\top, \tag{3.2}$$

where: $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_{N_R^{\mathbf{A}}}] \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_R^{\mathbf{A}}}$ and $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_{N_C^{\mathbf{A}}}] \in \mathbb{R}^{N_C^{\mathbf{A}} \times N_C^{\mathbf{A}}}$ are two orthogonal matrices, whose columns are the *left and right singular vectors* of $\mathbf{A}$, respectively; $\mathbf{\Sigma} \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_C^{\mathbf{A}}}$ is a pseudo-diagonal, semi-positive definite matrix with diagonal entries containing the *singular values* of $\mathbf{A}$, ordered as $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\mathscr{R}} \geq 0$, where $\mathscr{R} = \min(N_C^{\mathbf{A}}, N_R^{\mathbf{A}})$ is the rank of matrix $\mathbf{A}$.

Various algorithms are available for calculating the SVD of a matrix $\mathbf{A}$. One of them involves solving the following two eigenvalue problems:

$$\mathbf{A}\mathbf{A}^\top = \mathbf{P}\mathbf{\Sigma}\mathbf{\Sigma}^\top\mathbf{P}^\top, \tag{3.3}$$

$$\mathbf{A}^\top\mathbf{A} = \mathbf{Z}\mathbf{\Sigma}^\top\mathbf{\Sigma}\mathbf{Z}^\top. \tag{3.4}$$

222. Hotelling, *Analysis of a complex of statistical variables into principal components*. 1933

223. Pearson, *LIII. On lines and planes of closest fit to systems of points in space*. 1901

224. Kosambi, *Statistics in Function Space*. 1943

225. Mees *et al.*, *Singular-value decomposition and embedding dimension*. 1987

226. Liang *et al.*, *Proper orthogonal decomposition and its applications–Part I: Theory*. 2002

227. Wu *et al.*, *A note on equivalence of proper orthogonal decomposition methods*. 2003

124. Kerschen *et al.*, *The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview*. 2005

228. Kerschen *et al.*, *Physical interpretation of the proper orthogonal modes using the singular value decomposition*. 2002

229. Lenaerts *et al.*, *Identification of a continuous structure with a geometrical non-linearity. Part II: Proper orthogonal decomposition*. 2003

230. Gobat *et al.*, *Reduced order modeling of nonlinear microstructures through Proper Orthogonal Decomposition*. 2022

231. Corigliano *et al.*, *Domain decomposition and model order reduction methods applied to the simulation of multi-physics problems in MEMS*. 2013

232. Corigliano *et al.*, *Model Order Reduction and domain decomposition strategies for the solution of the dynamic elastic-plastic structural problem*. 2015

233. Guo *et al.*, *Reduced order modeling for nonlinear structural analysis using Gaussian process regression*. 2018

234. Tezzele *et al.*, *A multi-fidelity approach coupling parameter space reduction and non-intrusive POD with application to structural optimization of passenger ship hulls*. 2023

235. Corigliano *et al.*, *Coupled domain decomposition-proper orthogonal decomposition methods for the simulation of quasi-brittle fracture processes*. 2016

From the two spectral decompositions in Eq. (3.3) and Eq. (3.4), it follows that the l-th singular value of $\mathbf{A}$, with $l = 1 \ldots \mathscr{R}$, is the square root of the l-th eigenvalue of $\mathbf{A}^\top \mathbf{A}$*, and the following properties hold:

*If matrix $\mathbf{A}$ is symmetric, the singular values are the absolute value of its eigenvalues.

$$\|\mathbf{A}\|_2 = \sigma_1, \tag{3.5}$$

$$\|\mathbf{A}\|_F = \sqrt{\sigma_1^2 + \ldots + \sigma_{\mathscr{R}}^2}, \tag{3.6}$$

where, $\|\mathbf{A}\|_F \in \mathbb{R}$ is the *Frobenius norm* of $\mathbf{A}$, defined as:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{n_R=1}^{N_R^{\mathbf{A}}} \sum_{n_C=1}^{N_C^{\mathbf{A}}} a_{n_R,n_C}^2}, \tag{3.7}$$

with a_{n_R,n_C} being the entry of $\mathbf{A}$ at n_R-th row and n_C-th column. Additionally, the left singular vectors $\mathbf{p}_l$ and the right singular vectors $\mathbf{z}_l$ are the orthonormalized eigenvectors of $\mathbf{A}\mathbf{A}^\top$ and $\mathbf{A}^\top\mathbf{A}$, respectively. Hence, the POMs, defined as the eigenvectors of the sample covariance matrix† are equal to the left singular vectors of $\mathbf{A}$. Similarly, the eigenvalues of the sample covariance matrix coincide with the square of the singular values of $\mathbf{A}$ normalized by $N_C^{\mathbf{A}}$.

†The sample covariance matrix is identical to $\mathbf{A}\mathbf{A}^\top$ except for a normalization constant given by the number of observations.

Given a matrix $\mathbf{A} \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_C^{\mathbf{A}}}$ with $N_R^{\mathbf{A}} \geq N_C^{\mathbf{A}}$ and rank $\mathscr{R}$, its SVD is $\mathbf{A} = \mathbf{P}\boldsymbol{\Sigma}\mathbf{Z}^\top$. Hence, for $l = 1 \ldots \mathscr{R}$, the following relationships hold for each singular value σ_l:

$$\mathbf{A}\mathbf{z}_l = \sigma_l \mathbf{p}_l, \tag{3.8}$$

$$\mathbf{A}^\top \mathbf{p}_l = \sigma_l \mathbf{z}_l. \tag{3.9}$$

Thus, matrix $\mathbf{A}$ can be expressed as the sum of $\mathscr{R}$ 1-rank matrices:

$$\mathbf{A} = \sum_{l=1}^{\mathscr{R}} \sigma_l \mathbf{p}_l \mathbf{z}_l^\top, \tag{3.10}$$

allowing for a *low-rank approximation* of $\mathbf{A}$ by truncating the expansion to $N_{\mathrm{RB}}^{\mathbf{A}}$ POMs, with $0 < N_{\mathrm{RB}}^{\mathbf{A}} \leq \mathscr{R}$, as:

$$\mathbf{A}_{\mathrm{RB}} = \sum_{l=1}^{N_{\mathrm{RB}}^{\mathbf{A}}} \sigma_l \mathbf{p}_l \mathbf{z}_l^\top. \tag{3.11}$$

The optimality of Eq. (3.11) is ensured by the *Schmidt-Eckart-Young theorem* [236], stating that in the space $\mathbb{W}_{N_{\mathrm{RB}}^{\mathbf{A}}} = \{\mathbf{B} \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_{\mathrm{RB}}^{\mathbf{A}}} : \mathbf{B}^\top\mathbf{B} = \mathbf{I}_{\mathrm{RB}}\}$ of all possible $N_{\mathrm{RB}}^{\mathbf{A}}$-dimensional orthogonal bases in $\mathbb{R}^{N_R^{\mathbf{A}}}$, the basis matrix $\mathbf{W}_{\mathbf{A}} = [\mathbf{p}_1, \ldots, \mathbf{p}_{N_{\mathrm{RB}}^{\mathbf{A}}}] \in \mathbb{R}^{N_R^{\mathbf{A}} \times N_{\mathrm{RB}}^{\mathbf{A}}}$ minimizes the *projection error*

236. Eckart *et al.*, *The approximation of one matrix by another of lower rank.* 1936

occurring when approximating $\mathbf{A}$ by a matrix of rank $N_{\mathrm{RB}}^{\mathbf{A}}$, that is:

$$\begin{aligned}\min_{\mathbf{B}\in\mathbb{W}_{N_{\mathrm{RB}}^{\mathbf{A}}}} \|\mathbf{A}-\mathbf{B}\mathbf{B}^\top\mathbf{A}\|_F^2 = \|\mathbf{A}-\mathbf{W}_{\mathbf{A}}\mathbf{W}_{\mathbf{A}}^\top\mathbf{A}\|_F^2 = \\ = \sum_{n_C=1}^{N_C^{\mathbf{A}}} \|\mathbf{a}_{n_C}-\mathbf{W}_{\mathbf{A}}\mathbf{W}_{\mathbf{A}}^\top\mathbf{a}_{n_C}\|_2^2 = \\ = \sum_{l=N_{\mathrm{RB}}^{\mathbf{A}}+1}^{\mathscr{R}} \sigma_l^2.\end{aligned} \tag{3.12}$$

A similar result holds by considering the 2-norm:

$$\min_{\mathbf{B}\in\mathbb{W}_{N_{\mathrm{RB}}^{\mathbf{A}}}} \|\mathbf{A}-\mathbf{B}\mathbf{B}^\top\mathbf{A}\|_2^2 = \|\mathbf{A}-\mathbf{W}_{\mathbf{A}}\mathbf{W}_{\mathbf{A}}^\top\mathbf{A}\|_2^2 = \sigma_{N_{\mathrm{RB}}^{\mathbf{A}}+1}^2. \tag{3.13}$$

To complete the description of the SVD-based POD, we provide some possible interpretations of the POMs:

- *Geometric interpretation.* If the sample covariance matrix is real, symmetric and positive definite, then its eigenvectors represent the principal axes of an ellipsoid centered at the origin of the Euclidean space.

- *Energetic interpretation.* The SVD of a matrix $\mathbf{A}\in\mathbb{R}^{N_R^{\mathbf{A}}\times N_C^{\mathbf{A}}}$ provides crucial information about its *oriented energy* content. In this context, oriented energy refers to as the energy associated with the vectors forming $\mathbf{A}$, and is defined as:

$$\varepsilon(\mathbf{A}) = \|\mathbf{A}\|_F^2 = \sum_{n_R=1}^{N_R^{\mathbf{A}}}\sum_{n_C=1}^{N_C^{\mathbf{A}}} (a_{n_R,n_C})^2 = \sum_{l=1}^{\mathscr{R}} \sigma_l^2. \tag{3.14}$$

  The oriented energy associated with the vectors forming $\mathbf{A}$ in the direction of a unit vector $\mathbf{e}\in\mathbb{R}^{N_R^{\mathbf{A}}}$ is given by:

$$\varepsilon_{\mathbf{e}}(\mathbf{A}) = \sum_{n_C=1}^{N_C^{\mathbf{A}}} (\mathbf{e}^\top\mathbf{a}_{n_C})^2. \tag{3.15}$$

  An essential property of the SVD is that the maximum values in oriented energy occur in the left singular directions, coinciding with the square of the associated singular value. Accordingly, the POMs provide an optimal approximation, in the least squares sense, of the original space in terms of energy content. This property is crucial for understanding the number of POMs to be retained in the approximation, as described below.

- *Physical interpretation.* In the field of structural dynamics, the POMs can be physically interpreted, as they may resemble the physical mode shapes or even coincide with them in some particular cases. Since the SVD is signal-dependent, the POMs also depend on the system excitation. However, for linear, unforced, and undamped dynamic responses, the POMs converge to the mode shapes as $N_{\mathrm{C}}^{\mathbf{A}} \to \infty$ [237]. In the case of weakly damped structures, the POMs only approximate the mode shapes of the structure. Finally, if the excitation frequency matches a resonance frequency of the system, the dominant POM converges to the excited mode shape.

237. Feeny *et al.*, *On the physical interpretation of proper orthogonal modes in vibrations*. 1998

Eq. (3.12) quantifies the error occurred when approximating $\mathbf{A}$ using the POMs. Together with the energetic interpretation of the POMs, this result suggests the adoption of an *energy-based criterion* to select the order $N_{\mathrm{RB}}^{\mathbf{A}}$ of the POD-based approximation. As each singular value quantifies the information content described by the corresponding left singular vector, $N_{\mathrm{RB}}^{\mathbf{A}}$ can be set as the smallest integer such that the projection error is smaller than a prescribed tolerance $\epsilon_\sigma \in \mathbb{R}$ on the fraction of energy content to be disregarded in the approximation, according to:

$$\frac{\sum_{l=1}^{N_{\mathrm{RB}}^{\mathbf{A}}} \sigma_l^2}{\sum_{l=1}^{\mathscr{R}} \sigma_l^2} \geq 1 - \epsilon_\sigma^2, \tag{3.16}$$

that is, the oriented energy retained by the disregarded $\mathscr{R} - N_{\mathrm{RB}}^{\mathbf{A}}$ POMs is equal or smaller than ϵ_σ^2.

3.3.2 Reduced basis method in elasto-dynamics

In the following, we introduce the POD-Galerkin reduced basis method for parametrized elasto-dynamics problems, like (3.1).

To determine the POMs, it is necessary to gather information about the system to be approximated during a *training phase of the ROM*. This involves assembling a *snapshot matrix* $\mathbf{S} = [\mathbf{x}_1, \ldots, \mathbf{x}_{N_{\mathrm{S}}}] \in \mathbb{R}^{N_{\mathrm{FE}} \times N_{\mathrm{S}}}$ from $N_{\mathrm{S}} \in \mathbb{N}$ solution snapshots, representing time histories of nodal displacements computed by integrating Problem (3.1) over time for various parameter vectors $\boldsymbol{\mu}$. The parametric space underlying $\boldsymbol{\mu}$ is sampled using the Latin hypercube rule, and the adequacy of the number of snapshots N_{S} can be assessed a-posteriori by comparing the ROM and FOM outputs for unexplored parameter values. The matrix $\mathbf{S}$ is then factorized via SVD, and a POD basis matrix $\mathbf{W} \in \mathbb{R}^{N_{\mathrm{FE}} \times N_{\mathrm{RB}}}$, collecting $N_{\mathrm{RB}} \ll N_{\mathrm{FE}}$ POMs, is computed based on Eq. (3.16). Only after these preliminary operations, the ROM

can be assembled and solved in place of the FOM. The obtained ROM is specifically tailored to optimally represent the scenarios explored during training*. In this light, the POD offline phase is not only crucial for ensuring the quality of the reduction process but is also seen as a bridge between physics-based modeling and data-driven methods.

*In the presence of system alterations, significant changes in forcing characteristics, or nonlinearity, there might be a loss of representation capacity, requiring an update of the basis functions, or the introduction of local reduced bases

For any given set of input parameters $\boldsymbol{\mu}$, the FOM solution over the time interval $(0, T)$ consists of multiple frames tracking the temporal evolution of the system. In the context of POD, time can be treated as an additional parameter. However, it is not convenient to simultaneously deal with *temporal and parametric dependencies*, as this might complicate the algorithmic procedure and possibly hinder any physical interpretation of the POMs. We address this issue through the *two-step procedure* formalized in Algorithm 3.1. In detail:

I For each sampled parameter vector $\boldsymbol{\mu}_{n_P}$, $n_P = 1, \dots, N_P$, with $N_P \in \mathbb{N}$ being the number of samples drawn from the parameter space, we compute the FOM solution and store L snapshots in time† to create a first snapshot matrix $\mathbf{S}^{\text{time}}_{n_P} = [\mathbf{x}_L(\boldsymbol{\mu}_{n_P}) | \dots | \mathbf{x}_L(\boldsymbol{\mu}_{n_P})] \in \mathbb{R}^{N_{\text{FE}} \times L}$. A first POD basis matrix $\mathbf{W}^{\text{time}}_{n_P}$ is then obtained by performing an SVD on $\mathbf{S}^{\text{time}}_{n_P}$ and retaining a sufficient number of left singular vectors to satisfy Eq. (3.16) for a prescribed error tolerance ϵ_σ. By repeating this procedure for each sampled $\boldsymbol{\mu}_{n_P}$, we determine N_P different POD basis matrices.

†To speed up the offline phase, the collection of snapshots can be interrupted before solving the full-order model for the entire time interval $(0, T)$, provided that the gathered information on the dynamic evolution of the system allows for obtaining a steady-state solution in terms of basis functions (see, e.g., [238]).

II Once $\mathbf{W}^{\text{time}}_{n_P}$ is computed following stage I, for $n_P = 1, \dots, N_P$, we combine it with the previously identified basis set $\mathbf{W}$ to form a new snapshot matrix $\mathbf{S} = [\mathbf{W} | \mathbf{W}^{\text{time}}_{n_P}]$. Subsequently, a second SVD is performed on $\mathbf{S}$, and a sufficient number of left singular vectors is retained to satisfy Eq. (3.16)‡. This process progressively updates $\mathbf{W}$ to incorporate the parametric dependence of the problem.

‡It is worth noting that two different error tolerances ϵ_σ can be used to evaluate Eq. (3.16) for the time and parametric dependencies.

For any given set of input parameters $\boldsymbol{\mu}$, the POD-based ROM approximates the solution to Problem (3.1) as:

$$\mathbf{x}(t, \boldsymbol{\mu}) \approx \mathbf{W}\widehat{\mathbf{x}}(t, \boldsymbol{\mu}), \tag{3.17}$$

that is, a linear combination of $N_{\text{RB}} \ll N_{\text{FE}}$ POMs weighted by the *POD coefficients*, or *reduced DOFs* $\widehat{\mathbf{x}}(t, \boldsymbol{\mu}) \in \mathbb{R}^{N_{\text{RB}}}$.

Since the ROM solution and its time-derivatives do not satisfy the FOM problem, equality is restored through a *Galerkin projection* of the system. This is achieved by enforcing orthogonality between the residual of the FOM problem $\mathbf{f}(t, \boldsymbol{\mu}) - \mathbf{M}\mathbf{W}\ddot{\widehat{\mathbf{x}}}(t) - \mathbf{C}(\boldsymbol{\mu})\mathbf{W}\dot{\widehat{\mathbf{x}}}(t) - \mathbf{K}(\boldsymbol{\mu})\mathbf{W}\widehat{\mathbf{x}}(t)$, and the subspace

Algorithm 3.1 POD for parametrized elasto-dynamics

Input: N_{P} samples of $\boldsymbol{\mu}$
POD tolerance ϵ_σ

Solve FOM for $\boldsymbol{\mu}_1$
Collect $\mathbf{S}_1^{\mathrm{time}} = [\mathbf{x}_1(\boldsymbol{\mu}_1)| \dots |\mathbf{x}_L(\boldsymbol{\mu}_1)]$
$\mathbf{W} = \mathrm{POD}(\mathbf{S}_1^{\mathrm{time}})$
for $n_P = 2 : N_{\mathrm{P}}$ **do**
    Solve FOM for $\boldsymbol{\mu}_{n_P}$
    Collect $\mathbf{S}_1^{\mathrm{time}} = [\mathbf{x}_L(\boldsymbol{\mu}_{n_P})| \dots |\mathbf{x}_L(\boldsymbol{\mu}_{n_P})]$
    $\mathbf{W}_{n_P}^{\mathrm{time}} = \mathrm{POD}(\mathbf{S}_{n_P}^{\mathrm{time}})$
    $\mathbf{S} = [\mathbf{W}|\mathbf{W}_{n_P}^{\mathrm{time}}]$
    $\mathbf{W} = \mathrm{POD}(\mathbf{S})$
return POD basis matrix $\mathbf{W}$

spanned by the first N_{RB} POMs. Hence, we obtain the following N_{RB}-dimensional semi-discretized form:

$$\begin{cases} \mathbf{M}_r \ddot{\widehat{\mathbf{x}}}(t) + \mathbf{C}_r(\boldsymbol{\mu})\dot{\widehat{\mathbf{x}}}(t) + \mathbf{K}_r(\boldsymbol{\mu})\widehat{\mathbf{x}}(t) = \mathbf{f}_r(t, \boldsymbol{\mu}), \quad t \in (0, T) \\ \widehat{\mathbf{x}}(0) = \mathbf{W}^\top \mathbf{x}_0, \\ \dot{\widehat{\mathbf{x}}}(0) = \mathbf{W}^\top \dot{\mathbf{x}}_0. \end{cases} \tag{3.18}$$

Here, the reduced-order matrices $\mathbf{M}_r \in \mathbb{R}^{N_{\mathrm{RB}} \times N_{\mathrm{RB}}}$, $\mathbf{C}_r \in \mathbb{R}^{N_{\mathrm{RB}} \times N_{\mathrm{RB}}}$, and $\mathbf{K}_r \in \mathbb{R}^{N_{\mathrm{RB}} \times N_{\mathrm{RB}}}$, and vector $\mathbf{f}_r \in \mathbb{R}^{N_{\mathrm{RB}}}$ play the same role as their FOM counterparts, but with dimensions $N_{\mathrm{RB}} \times N_{\mathrm{RB}}$ instead of $N_{\mathrm{FE}} \times N_{\mathrm{FE}}$, and read:

$$\mathbf{M}_r \equiv \mathbf{W}^\top \mathbf{M} \mathbf{W}, \tag{3.19}$$

$$\mathbf{C}_r(\boldsymbol{\mu}) \equiv \mathbf{W}^\top \mathbf{C}(\boldsymbol{\mu}) \mathbf{W}, \tag{3.20}$$

$$\mathbf{K}_r(\boldsymbol{\mu}) \equiv \mathbf{W}^\top \mathbf{K}(\boldsymbol{\mu}) \mathbf{W}, \tag{3.21}$$

$$\mathbf{f}_r(t, \boldsymbol{\mu}) \equiv \mathbf{W}^\top \mathbf{f}(t, \boldsymbol{\mu}). \tag{3.22}$$

The solution of the low-dimensional dynamical system in Problem (3.18) is advanced in time using the same strategy employed for the FOM and then traced back to the original FOM space according to Eq. (3.17). Subsequently, we can post-process the approximated solution to extract synthetic vibration recordings emulating a sensing system. In particular, vibration recordings shaped as multivariate time series $\mathbf{U}(\boldsymbol{\mu}) = [\mathbf{u}_1(\boldsymbol{\mu}), \dots, \mathbf{u}_{N_u}(\boldsymbol{\mu})] \in \mathbb{R}^{L \times N_u}$, consisting of N_u series with L measurements equally spaced in time, in terms of accelerations or displacements, are obtained as follows.

With reference to displacement recordings, nodal values over $(0, T)$ are first collected as $\mathbf{V} = [\mathbf{W}\widehat{\mathbf{x}}_1, \dots, \mathbf{W}\widehat{\mathbf{x}}_L] \in \mathbb{R}^{N_{\mathrm{FE}} \times L}$ by solving Problem (3.18). The relevant vibration recordings forming $\mathbf{U}$ are then extracted as:

$$\mathbf{U} = (\mathbf{T}\mathbf{V})^\top, \tag{3.23}$$

where $\mathbf{T} \in \mathbb{B}^{N_u \times N_{\text{FE}}}$ is a Boolean matrix whose (n, m)-th entry is equal to 1 only if the n-th sensor output coincides with the m-th DOF. It is worth noting that the matrix product $\mathbf{TW} \in \mathbb{R}^{N_u \times N_{\text{RB}}}$ can be computed, once and for all, to extract $\mathbf{U}$ for any given set of input parameters $\boldsymbol{\mu}$. In order to mimic the measurement noise, each vibration recording can be corrupted by adding an *independent, identically distributed* Gaussian noise, with statistical properties depending on the target accuracy of the sensors.

3.4 THESIS PERSPECTIVE

In this Chapter, we have discussed a strategy that combines the parametrization of damage and operational conditions with the reduced basis method in elasto-dynamics. The achieved reduced-order modeling framework allows for the efficient simulation of structural responses, preserving the explicit dependence of the solution on input parameters and time while controlling the level of introduced approximation. This strategy provides the foundation for the simulation-based SHM paradigm employed throughout this Thesis.

By providing a knowledge platform capable of replicating the impact of damage on structural response under varying operational conditions, the considered numerical models are systematically employed to efficiently populate training datasets:

I Referring to the damage localization framework proposed in Chapter 5, the DL model underlying the extraction of informative features from raw vibration recordings is trained in a supervised pairwise fashion. This involves leveraging synthetic data from ROM simulations under varying operational and damage conditions. The same training data, suitably post-processed, are then used to train a k-NN regressor applied on top of the feature extractor for real-time damage localization.

II In Chapter 6, we show how to reduce the computational cost of supervised training while ensuring the accuracy of the approximated signals within the proposed MF-DNN surrogate modeling strategy. In Sec. 6.1, we describe the sequential training of the LF and HF components comprising the MF-DNN-SM. This involves considering a large amount of LF data generated using an LF-ROM and a small amount of HF data generated through an HF-FE model, respectively, in the absence or presence of structural damage. The parameters describing the location and the amplitude of damage are all addressed from a regression perspective,

despite there are no restrictions in this respect. Similar to point I, in Sec. 6.3, we also demonstrate how to use the same HF data to train a feature extractor and a k-NN regressor for the identification of operational conditions.

III Within the enhanced Bayesian model updating framework proposed in Chapter 7, the feature extractor and the feature-oriented SM are also learned relying on simulation-based supervision. However, in this case, training data are cost-effectively generated through the MF-DNN-SMs obtained in Chapter 6.

IV Within the digital twinning framework proposed in Chapter 8, the parametrization of damage and operational conditions is combined with reduced order modeling to efficiently assemble training datasets, similarly to Chapter 5. However, adopting a slightly different perspective from the rest of this Thesis, the structural models are parametrized to describe damage scenarios within a finite set of discrete classes, each associated with a predefined damage region. Accordingly, the related damage localization problem is formulated from a classification perspective, rather than a regression one.

This page intentionally left blank.

4 MACHINE LEARNING FOR PREDICTIVE DIGITAL TWINS

In this Chapter, we focus on ML methods, envisioned to provide the algorithmic platform that supports physics-based modeling for realizing effective DTs of engineering structures. Overall, ML comprises a collection of methods designed to learn relationships between data and quantities of interest. The primary goal is to gain insights into the behavior of a system and make predictions for unobserved scenarios. ML shifts the role of engineers from manual learning to programming a computer for *automated learning*. The learning process involves tuning a predictive model to optimize its performance in predicting known target quantities.

ML and artificial intelligence are closely interconnected. In essence, artificial intelligence aims to perceive and dynamically interact with the real-world by making rational decisions*. Behind such an artificial intelligence system there are ML methods that extract information from observations, predict future system responses, and determine the optimal course of actions.

By examining the SHM problem through the ML lens, the pattern recognition paradigm for SHM [71] encompasses the following steps: (i) operational evaluation; (ii) data acquisition; (iii) feature selection and extraction; (iv) statistical modeling to unveil the relationship between the selected features and sought damage patterns. Although operational evaluation and data acquisition, introduced in Sec. 1.2.3, serve as essential starting points for any SHM strategy, this Chapter focuses on feature selection/extraction and statistical modeling.

The Chapter is organized as follows. Sec. 4.1 is devoted to introducing key aspects of ML, covering the rationale behind regression and classification algorithms, the concepts of *training*, *validation* and *testing* of a learnable model, and the introduction of the increasingly popular DL paradigm. The DL paradigm is then extensively analyzed in Sec. 4.2, focusing on *function approximation* capabilities, as well as *optimization* and *regularization* techniques for DL models. The DL architectures employed in this Thesis, such as *convolutional neural networks* (CNNs), *recurrent neural networks* (RNNs), and *autoencoders* (AEs), are also

Part of the research presented in this Chapter has been published as:

[164] *Torzoni et al., Structural health monitoring of civil structures: A diagnostic framework powered by deep metric learning,* Computers & Structures, *Vol. 271, 2022, p. 106858.*

[165] *Torzoni et al., A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks,* Mechanical Systems and Signal Processing, *Vol. 197, 2023, p. 110376.*

[166] *Torzoni et al., Enhancing Bayesian model updating in structural health monitoring via learnable mapping, 2024.* Under review.

[239] *Torzoni et al., A digital twin framework for civil engineering structures,* Computer Methods in Applied Mechanics and Engineering, *Vol. 418, 2024, p. 116584.*

[167] *Torzoni et al., 5 Damage identification using physics-based datasets: From convolutional to metric-informed damage-sensitive feature extractors,* Data-Centric Structural Health Monitoring: Mechanical, Aerospace and Complex Infrastructure Systems, De Gruyter, *2023, pp. 101–124.*

*We highlight how this broad-sense definition of artificial intelligence clearly epitomizes the essence of digital twins.

71. Farrar *et al., Vibration–Based Structural Damage Identification.* 2001

discussed in detail in Sec. 4.2, highlighting the peculiarities that make them suitable for SHM purposes. Finally, Sec. 4.3 provides a detailed explanation of the conceptual and theoretical underpinnings behind *dynamic Bayesian networks* and *dynamic decision networks*, employed in this Thesis for the digital twinning of engineering systems.

4.1 KEY ASPECTS OF MACHINE LEARNING

ML algorithms use synthetic representation of the data, known as features, that are ideally informative for making predictions about unobserved variables of interest. In SHM, damage-sensitive features are employed to distill high-dimensional sensor data, capturing meaningful information for damage identification while mitigating the impact of operational and environmental variabilities. In this context, statistical models serve to establish a mapping between input features and target variables, describing potential damage scenarios. ML predictions are typically refined by an optimization algorithm that leverages training data referring to undamaged or, possibly, damaged states of the structure. The learned model is then deployed to provide informed predictions by processing data features extracted from previously unseen data.

The learning process to train a statistical model can be of different types. For our purposes, three primary ML sub-fields can be identified: *supervised* learning, *unsupervised* learning, and *reinforcement* learning*. *Supervised learning* is employed when building a model to describe the relationships between observed system responses and sought characteristics or properties of the system, given that these latter are known when learning the statistical model. *Unsupervised learning*, on the other hand, aims to discover structures, patterns, or anomalies without knowledge of the correct answers, as the target outputs are not employed during training. While supervised methods in SHM exploit training data corresponding to the baseline undamaged condition and to known damage scenarios that may affect the structure, unsupervised ones rely solely on unlabeled training data, typically referring to the baseline undamaged condition. Lastly, *reinforcement learning* focuses on learning a *control policy* mapping each possible state of a system onto suitable actions to maximize the *long-term accumulation of rewards*.

*It is important to note that this classification of machine learning methods shows some limitations, as there is often an overlap between sub-fields and, sometimes, unclear distinctions due to shared mathematical formulations.

4.1.1 REGRESSION PERSPECTIVE

The choice of the type of statistical model is influenced not only by the type of selected features but also by the specific task to

be addressed, such as damage detection, localization, and quantification. In the following two Sections, we discuss potential problems typical of SHM and how these can be addressed from supervised regression and classification perspectives. A typical unsupervised setting is instead discussed in Sec. 4.2.7, dealing with AE-based dimensionality reduction.

Regression involves modeling the relationships between data features and the sought properties of a system in a *non-discretized setting* [196]*. Popular algorithms include logistic regression [70], support vector machines [144], Gaussian processes regression [240], and neural networks [103], among the others.

196. Goulet, *Probabilistic Machine Learning for Civil Engineers*. 2020

*The non-discretized setting here refers to the value range of the quantities of interest to be predicted with the regression model, not to be confused with the finite element discretization characterizing the physiscs-based models described in Chapter 3.

70. Bishop, *Pattern Recognition and Machine Learning*. 2006

144. Gunn, *Support Vector Machines for Classification and Regression*. 1998

240. Easmussen, *Gaussian Processes in Machine Learning*. 2004

103. Goodfellow *et al.*, *Deep Learning*. 2016

In the context of SHM, the regression perspective is well-suited for tasks such as damage localization and damage quantification. While for damage localization the data features are mapped onto a set of parameters describing the coordinates of a damage region, when exploited for damage quantification purposes, the data features are instead mapped onto the severity of damage. On the other hand, the problem of damage detection is almost never addressed from a regression perspective, as the presence or absence of damage in a structure can be more naturally assessed from a classification perspective. Referring to the modeling choice introduced in Sec. 3.2, the quantities of interest for regression include the spatial coordinates of the applied stiffness reduction, denoted as $\mathbf{y} \in \mathbb{R}^3$, and the associated magnitude $\delta \in \mathbb{R}$, both included in the vector $\boldsymbol{\mu} \in \mathbb{R}^{N_{\text{par}}}$ describing the variability of the monitored system.

For our purposes, the dataset to be used for training a regressor to address the problem of damage localization or quantification can be first defined as:

$$\mathcal{D}_{\text{RG}} = \{(\mathbf{U}_i, \mathbf{y}_i, \delta_i)\}_{i=1}^{N_{\text{train}}}, \tag{4.1}$$

where synthetic vibration recordings $\mathbf{U}_i \in \mathbb{R}^{L \times N_u}$ are generated by simulating the structural response for the i-th sampling of $\boldsymbol{\mu}_i$, with $i = 1, \ldots, N_{\text{train}}$, and labeled by the corresponding values of $\mathbf{y}$ and δ, according to the procedure detailed in Sec. 3.3.2.

Before training, $\mathcal{D}_{\text{RG}}$ has to be post-processed to extract informative features $\mathbf{h} \in \mathbb{R}^{N_h}$ to feed the regressor. By mapping the vibration recordings in $\mathcal{D}_{\text{RG}}$ onto the selected feature space, a second training dataset is derived from $\mathcal{D}_{\text{RG}}$ as:

$$\mathcal{D}_{\text{RG}}^{\mathbf{h}} = \{(\mathbf{h}_i, \mathbf{y}_i, \delta_i)\}_{i=1}^{N_{\text{train}}}. \tag{4.2}$$

The dataset in Eq. (4.2) is eventually exploited to train a regressor $\mathcal{M}_{\text{RG}}$ to addresses one of the regression tasks underlying the damage localization and damage quantification problems, namely $\mathcal{M}_{\text{RG}}^{\mathbf{y}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}^3$, $\mathbf{h} \mapsto \mathbf{y}$, and $\mathcal{M}_{\text{RG}}^{\delta} : \mathbb{R}^{N_h} \mapsto \mathbb{R}$, $\mathbf{h} \mapsto \delta$.

By denoting the query data features extracted from incoming vibration recordings $\mathbf{U}^{\text{EXP}}$ as $\mathbf{h}^{\text{EXP}}$, the estimated counterparts of damage location $\mathbf{y}$ and associated magnitude δ are obtained as $\widehat{\mathbf{y}} = \mathcal{M}^{\mathbf{y}}_{\text{RG}}(\mathbf{h}^{\text{EXP}})$ and $\widehat{\delta} = \mathcal{M}^{\delta}_{\text{RG}}(\mathbf{h}^{\text{EXP}})$, respectively.

An instance of $\mathcal{M}^{\mathbf{y}}_{\text{RG}}$ considered in this Thesis is the k-NN regressor exploited in Chapter 5 for damage localization. An instance of $\mathcal{M}^{\delta}_{\text{RG}}$ is instead the DL regressor for damage quantification used in Chapter 8 within the proposed DT framework, which also addresses the feature extraction task. Finally, deriving $\mathcal{D}^{\mathbf{h}}_{\text{RG}}$ from $\mathcal{D}_{\text{RG}}$ can be likened to the DL models for feature extraction exploited in Chapter 5, Sec. 6.3, and Chapter 7, all relying on a *Siamese architecture*.

Another situation of interest that is suitable to the regression perspective is surrogate modeling. In contrast to the previous case, surrogate modeling involves reversing the direction of the mapping. It entails building an emulator that mimics the input-output map, going from input parameters to sensor data or relevant data features. SMs are extensively used in SHM to replace computationally expensive physics-based models when estimating SHM parameters using model-based approaches. With reference to the numerical models introduced in Chapter 3, a dataset for training a SM can be assembled as:

$$\mathcal{D}_{\text{SM}} = \{(\boldsymbol{\mu}_i, \mathbf{U}_i)\}_{i=1}^{N_{\text{train}}}, \tag{4.3}$$

where the i-th set of input parameters $\boldsymbol{\mu}_i$ is labeled by the corresponding vibration recordings $\mathbf{U}_i$, with $i = 1, \dots, N_{\text{train}}$.

Even if the input and output quantities collected in $\mathcal{D}_{\text{SM}}$ are formally different from those employed in $\mathcal{D}_{\text{RG}}$, the training and evaluation workflow of a SM does not change compared to that of $\mathcal{M}^{\mathbf{y}}_{\text{RG}}$ or $\mathcal{M}^{\delta}_{\text{RG}}$. In particular, $\mathcal{D}_{\text{SM}}$ serves to train a SM $\mathcal{M}_{\text{SM}}$ to address the regression task $\mathcal{M}_{\text{SM}} : \mathbb{R}^{N_{\text{par}}} \mapsto \mathbb{R}^{L \times N_u}$, $\boldsymbol{\mu} \mapsto \mathbf{U}$. By denoting the query input parameters as $\boldsymbol{\mu}^{\text{EXP}}$, the trained $\mathcal{M}_{\text{SM}}$ provides the estimated counterpart of $\mathbf{U}$ as $\widehat{\mathbf{U}} = \mathcal{M}_{\text{SM}}(\boldsymbol{\mu}^{\text{EXP}})$. The same workflow applies for a SM mapping input parameters onto relevant data-features.

An instance of $\mathcal{M}_{\text{SM}}$ considered in this Thesis is represented by the MF-DNN-SMs derived in Chapter 6. On the other hand, the scenario of a SM mapping input parameters onto relevant data-features is well-captured by the feature-oriented SM exploited in Chapter 7 to enhance Bayesian model updating.

4.1.2 CLASSIFICATION PERSPECTIVE

While the target quantities in regression tasks are continuous variables, with *classification*, predictions are provided in a *discretized setting*, targeting discrete *classes*. Common classification

algorithms include generative classifiers [196], logistic regression [70], support vector machines [144], decision trees [241], Gaussian process classification [240], and neural networks [103].

196. Goulet, *Probabilistic Machine Learning for Civil Engineers*. 2020

70. Bishop, *Pattern Recognition and Machine Learning*. 2006

144. Gunn, *Support Vector Machines for Classification and Regression*. 1998

241. Quinlan, *Induction of decision trees*. 1986

240. Easmussen, *Gaussian Processes in Machine Learning*. 2004

103. Goodfellow *et al.*, *Deep Learning*. 2016

In the context of SHM, the classification perspective is well-suited for tasks such as damage detection and damage localization, where data features are categorized into predefined output classes, each one referring to a different damage scenario. On the other hand, the problem of damage quantification can be more naturally addressed within a non-discretized setting. When modeling structural damage as a local reduction in effective stiffness, and the problem of damage detection and localization is addressed from a classification perspective, the parameter $\mathscr{d} \in \{0, \dots, N_{\mathscr{d}}\}$ represents a specific damage scenario, among a set of $N_{\mathscr{d}}$ predefined damage locations*.

*We remark that $\mathscr{d} = 0$ denotes the damage-free baseline.

The dataset for training a classifier to address the problem of damage detection and localization is initially defined as:

$$\mathcal{D}_{\mathrm{CL}} = \{(\mathbf{U}_i, \mathbf{b}_i)\}_{i=1}^{N_{\mathrm{train}}}, \tag{4.4}$$

where each set of vibration recordings $\mathbf{U}_i$, with $i = 1, \dots, N_{\mathrm{train}}$, is labeled by a Boolean vector $\mathbf{b}_i \in \mathbb{B}^{N_{\mathscr{d}}+1}$ — commonly referred to as *one-hot encoding* — categorizing one of the $N_{\mathscr{d}} + 1$ predefined damage scenarios described through parameter $\mathscr{d}$. One-hot encoding is used instead of directly employing $\mathscr{d}$ because not all classification models can operate on nominal data directly. The one-hot encoding transforms the nominal feature described by $\mathscr{d}$ into a multi-dimensional binary vector. The number of dimensions corresponds to the number of categories, and each dimension corresponds to one category. Each category is represented by a vector in which the entry corresponding to the category's dimension is set to 1, and all other entries are set to 0.

Similar to the regression case, $\mathcal{D}_{\mathrm{CL}}$ is post-processed to provide a second training dataset as:

$$\mathcal{D}_{\mathrm{CL}}^{\mathbf{h}} = \{(\mathbf{h}_i, \mathbf{b}_i)\}_{i=1}^{N_{\mathrm{train}}}, \tag{4.5}$$

collecting feature representations $\mathbf{h}_i$, labeled by the associated one-hot encoding $\mathbf{b}_i$, with $i = 1, \dots, N_{\mathrm{train}}$.

The dataset in Eq. (4.5) is eventually employed to train a classifier $\mathcal{M}_{\mathrm{CL}}$ to addresses the *multi-class* classification task underlying the damage detection and localization problem, namely $\mathcal{M}_{\mathrm{CL}} : \mathbb{R}^{N_h} \mapsto \mathbb{B}^{N_{\mathscr{d}}+1}, \mathbf{h} \mapsto \mathbf{b}$. Once trained, $\mathcal{M}_{\mathrm{CL}}$ is used to provide the estimated counterpart of $\mathbf{b}$ as $\widehat{\mathbf{b}} = \mathcal{M}_{\mathrm{CL}}(\mathbf{h}^{\mathrm{EXP}})$, for any given query data features $\mathbf{h}^{\mathrm{EXP}}$.

While the regression and classification perspectives have been introduced separately, they can be conveniently combined and jointly exploited. For instance, a practical setup for detecting,

locating and quantifying the presence of damage could be as follows, analogous to that employed in Sec. 8.2 within the proposed digital twinning framework. Every time new observational data are acquired, they are first processed with a classification model $\mathcal{M}_{\mathrm{CL}}$ to address damage detection and localization. Whenever damage is detected and localized, the observational data are further processed with a regression model $\mathcal{M}_{\mathrm{RG}}^{\delta}$ — one for each damageable region — to estimate the magnitude of the stiffness reduction within the respective damage region.

4.1.3 Quality and quantity of training data

The number of training data required for satisfactory performance depends on the model complexity and the difficulty of the task to be addressed. Increasing N_{train} generally leads to a better statistical model. However, beyond a model/problem-dependent threshold, increasing N_{train} only produce increasingly subtle performance improvements. If the complexity of the model is excessive compared to N_{train}, the associated risk is that of a too-flexible model that overfit the training data.

Overfitting happens when the selected model performs the training task correctly, but fails to provide good predictions for unseen scenarios. In such cases, the model essentially memorizes the training data without truly catching the underlying relationships. Overfitting corresponds to a situation where the *prediction error* displays a low bias when evaluated against the training data; however, when validated with data unseen during training, the prediction error displays a high variance. Optimal model complexity involves a trade-off between the prediction bias and variance. Consistently with the *principle of Occam's razor*, it is crucial to keep the model as simple as possible.

A general procedure to identify the optimal model complexity is to employ a *cross-validation strategy*, as detailed in Sec. 2.2. An alternative approach to address this challenge is to optimize the learnable parameters from a Bayesian perspective and then select the appropriate model complexity via *Bayesian model class selection* [242]. Another viable approach is to regularize the learnable parameters, for instance via *ridge regression* [243].

242. Beck *et al.*, *Model Selection Using Response Measurements: Bayesian Probabilistic Approach.* 2004

243. Hoerl *et al.*, *Ridge regression: Biased estimation for nonorthogonal problems.* 1970

4.1.4 Validation and testing

To assess the generalization capability of a trained model, it is essential to evaluate its performance on samples that were not part of the training data. Conventionally, this is achieved by partitioning the dataset into two distinct sets: the *training set* and the *test set*. The model undergoes training using the training

set, while its generalization ability is assessed on the test set. Depending on the employed ML algorithm, it may be desirable to extract a subset of instances from the training set to form a *validation set*. Although not directly used for tuning the learnable parameters of the model, the validation set serves to assess the model performance and potentially halt the training process if overfitting is detected. Typically, the model optimization is carried out by considering an 80 : 20 splitting ratio for training and validation purposes. Therefore, 20% of the instances from the training set are randomly taken and set aside to validate the learning process. This validation involves assessing the values of a performance metric attained on the validation set.

4.1.5 DEEP LEARNING

While conventional ML techniques have limitations when dealing with data in its raw form, transforming raw data into informative features requires substantial domain expertise. DL models are *representation learning** methods exploiting the composition of multiple nonlinear modules, or *processing layers*. These layers transform the representation at one level, starting with the raw input, into a slightly more abstract representation at a higher level. Higher layers of representation amplify aspects of the input that are important for the task at hand and suppress other variations. The key feature of DL lies in the fact that these layers are not manually designed, but learned from data using the *backpropagation method* [244,245]. This method serves as a general-purpose learning procedure, guiding the adjustment of the learnable parameters in each layer based on the output of the previous layer [102].

*"Representation learning" is a branch of ML focusing on learning data representations.

244. Rumelhart *et al.*, *Learning Internal Representations by Error Propagation*. 1987

245. Rumelhart *et al.*, *Learning representations by back-propagating errors*. 1986

102. LeCun *et al.*, *Deep learning*. 2015

4.2 NEURAL NETWORKS

The fundamental idea that inspired neural networks (NNs) was to mimic the human nervous system. The initial computational model, simulating the interaction of multiple biological neurons and serving as the first example of an *artificial neuron*, is the *McCulloch-Pitts model* introduced in 1943 [246]. A more powerful and versatile computational model of the artificial neuron was proposed by Rosenblatt in 1958, known as the *perceptron* [247], which later became the reference basic unit of NNs.

246. McCulloch *et al.*, *A logical calculus of the ideas immanent in nervous activity*. 1943

247. Rosenblatt, *The perceptron: A probabilistic model for information storage and organization in the brain*. 1958

The perceptron is a linear threshold unit that produces a single output through an *affine transformation* of an input vector ruled by a nonlinear *activation function*, such as a step function or sigmoid function. In the perceptron model, the parameters governing the affine transformation of the input are not fixed

but learned from data*. A DL architecture is essentially a stack of simple layers, most of which undergo learning.

NNs are frequently acknowledged as universal approximators of continuous functions [248] and continuous operators [249]. In addition to their flexible approximation capabilities, NNs are recognized for their ease of use, facilitated by user-friendly application programming interfaces, such as the `Tensorflow`-based `Keras` [250]. Moreover, they are exempt from the curse of dimensionality, making them well-suited for high-dimensional problems. NNs are also cost-effective, fast to evaluate, and easily parallelizable. However, NNs come with several drawbacks, including their limited extrapolation capabilities, their low interpretability, and their need of large training datasets.

The most appealing aspect of NNs in an SHM context is their ability to automatically learn the selection and extraction of damage-sensitive features from raw data — a key capability given the typically low sensitivity of sensor recordings to damage. Optimized damage-sensitive features are selected and extracted to address the subsequent statistical modeling task through an end-to-end learning process [104]. However, the main limitation of DL-extracted features is their low interpretability. While engineered features typically enjoy a clear physical meaning, DL features are often challenging to interpret.

The remainder of this Section is organized as follows. In Sec. 4.2.1, we present the simplest NN architecture, the *feed-forward fully-connected* model. In Sec. 4.2.2, we show how to establish a regression or classification model by changing the type of *output nonlinearity* employed in the last layer of the DL model. Techniques for the optimization and regularization of DL models are described in Sec. 4.2.3 and in Sec. 4.2.4, respectively. Finally, we present the state-of-the-art DL models employed in this Thesis, encompassing CNNs in Sec. 4.2.5, RNNs in Sec. 4.2.6, and AEs in Sec. 4.2.7, respectively. References for the equations employed in the following can be found in [103].

*Originally, the perceptron learning algorithm was based on a simple Hebbian learning rule. However, a significant shift on the reference training procedure occurred with the introduction of the back-propagation method [244, 245], enabling to scale these simple models to deep neural networks.

248. Cybenko, *Approximation by superpositions of a sigmoidal function*. 1989

249. Chen *et al.*, *Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems*. 1995

250. Chollet *et al.*, *Keras*. 2015

104. Fink *et al.*, *Potential, Challenges and Future Directions for Deep Learning in Prognostics and Health Management Applications*. 2020

103. Goodfellow *et al.*, *Deep Learning*. 2016

4.2.1 FULLY-CONNECTED LAYER

In a fully-connected layer, each neuron is densely connected to its input layer. The computational model underlying a fully-connected layer is presented with reference to the ℓ-th fully-connected layer $\mathcal{F}^\ell : \mathbb{R}^{N_N^{\ell-1}} \mapsto \mathbb{R}^{N_N^\ell}$, $\boldsymbol{a}^{\ell-1} \mapsto \boldsymbol{a}^\ell$, with $\ell = 1, \dots, N_\ell$, in a DNN featuring $N_\ell \in \mathbb{N}$ layers (refer to Fig. 4.1). Denoting by $N_N^\ell \in \mathbb{N}$ the number of neurons in $\mathcal{F}^\ell$, each of them provides a scalar output $a_n^\ell \in \mathbb{R}$, with $n = 1, \dots, N_N^\ell$, computed as an affine transformation of its input

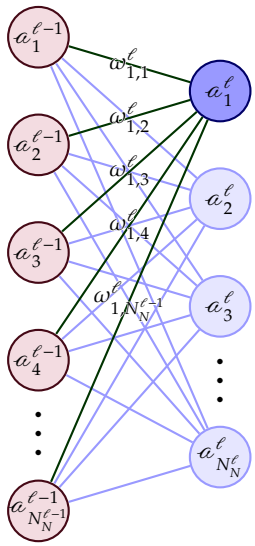


Figure 4.1: Exemplary fully-connected layer.

$\boldsymbol{a}^{\ell-1}$, ruled by an activation function $\mathcal{A} : \mathbb{R} \mapsto \mathbb{R}$ as follows:

$$a_n^{\ell} = \mathcal{F}_n^{\ell}(\boldsymbol{a}^{\ell-1}) = \mathcal{A}(\boldsymbol{\omega}_n^{\ell} \cdot \boldsymbol{a}^{\ell-1} + \beta_n^{\ell}). \tag{4.6}$$

In Eq. (4.6), $\boldsymbol{\omega}_n^{\ell} \in \mathbb{R}^{N_N^{\ell-1}}$ collects the *connection weights* governing the linear mapping for the n-th neuron of $\mathcal{F}^{\ell}$, and $\beta_n^{\ell} \in \mathbb{R}$ is the corresponding *bias term*. During each *forward pass*, the output of $\mathcal{F}^{\ell}$ is collectively assembled by aggregating the computations of its N_N^{ℓ} neurons as $\boldsymbol{a}^{\ell} = (a_1^{\ell}, \dots, a_{N_N^{\ell}}^{\ell})^{\top}$. The relationship in Eq. (4.6) is learnable through the matrices of tunable connection weights $\boldsymbol{\Xi}^{\ell} = [\boldsymbol{\omega}_1^{\ell}, \dots, \boldsymbol{\omega}_{N_N^{\ell}}^{\ell}] \in \mathbb{R}^{N_N^{\ell} \times N_N^{\ell-1}}$ and biases $\boldsymbol{\beta}^{\ell} = (\beta_1^{\ell}, \dots, \beta_{N_N^{\ell}}^{\ell})^{\top} \in \mathbb{R}^{N_N^{\ell}}$ associated with the affine transformations computed by each of the N_N^{ℓ} neurons in $\mathcal{F}^{\ell}$. The weights $\boldsymbol{\Xi}^{\ell}$ and biases $\boldsymbol{\beta}^{\ell}$ are tuned through the optimization of a task-dependent *loss function*, as detailed in Sec. 4.2.3.

By stacking two fully-connected layers, information is transferred from the input layer to the output layer by propagating it through a layer of hidden neurons. The resulting *multi-layer perceptron* architecture is referred to as a feed-forward NN with a single *hidden layer*. The output of the first layer is passed to the second layer, creating an *internal representation* of the inputs — essentially, a set of features. Stacking multiple layers in this way allows for strong nonlinear transformations of the input, enabling the approximation of highly intricate target functions.

4.2.2 Last layer activation function

An activation function describes a nonlinear transformation of a hidden variable. Common activation functions are the logistic,

the hyperbolic tangent (Tanh), the rectified linear unit (ReLU), and many other variants. The choice of the activation function for the output layer is crucial for the effectiveness of a NN.

In regression tasks, it is customary not to employ any activation function in the last layer. In this way, hidden layers handle the nonlinear aspect of the problem, and the output layer simply apply a linear mapping to retrieve the range of target values. With this setting, common loss functions for assessing the NN performance during training include the MAE in Eq. (2.17), the RMSE in Eq. (2.18), and the *mean squared error* (MSE), defined as:

$$\text{MSE} = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \|\mathbf{y}_i - \widehat{\mathbf{y}}_i\|_2^2, \tag{4.7}$$

whose minimization can be interpreted as maximizing the likelihood of the training dataset on the parameters of the model. Here, $\mathbf{y}_i$ and $\widehat{\mathbf{y}}_i$, for $i = 1, \ldots, N_{\text{train}}$, represent the actual and predicted values, respectively, in the context of the regression task underlying the damage localization problem discussed in Sec. 4.1.1. In this scenario, a DL model $\text{NN}^{\mathbf{y}}_{\text{RG}}$ is trained using the dataset in Eq. (4.1) to approximate the target mapping $\text{NN}^{\mathbf{y}}_{\text{RG}} : \mathbb{R}^{L \times N_u} \mapsto \mathbb{R}^3$, $\mathbf{U} \mapsto \mathbf{y}$. Once trained, it provides estimated values for $\mathbf{y}$ as $\widehat{\mathbf{y}} = \text{NN}^{\mathbf{y}}_{\text{RG}}(\mathbf{U}^{\text{EXP}})$. This setting also applies to the damage quantification problem discussed in Sec. 4.1.1.

In contrast, for binary classification tasks, a Tanh or logistic activation function can be used as the activation for the last layer, while Softmax is the standard choice for multi-class classification problems. For example, in the multi-class classification task underlying the damage localization problem discussed in Sec. 4.1.2, a DL model NN_{CL} can be trained using the dataset $\mathcal{D}_{\text{CL}}$ in Eq. (4.4) to approximate the target mapping $\text{NN}_{\text{CL}} : \mathbb{R}^{L \times N_u} \mapsto \mathbb{B}^{N_\ell + 1}$, $\mathbf{U} \mapsto \mathbf{b}$. In this case, the loss function typically used to guide the learning process is the probabilistic *categorical cross-entropy* between the predicted and the target class labels over $\mathcal{D}_{\text{CL}}$, which is defined as:

$$\mathcal{H} = -\frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \sum_{n=0}^{N_\ell} b_i^n \log(\widehat{b}_i^n), \tag{4.8}$$

providing a measure of the dissimilarity between the discrete probability distribution describing $\mathbf{b}$ and its estimated counterpart $\widehat{\mathbf{b}}$, obtained through NN_{CL}. By employing a Softmax activation for the output layer of NN_{CL}, the entries of $\widehat{\mathbf{b}} = (\widehat{b}^0, \ldots, \widehat{b}^{N_\ell})^\top$ are interpreted as confidence levels by which $\mathbf{U}^{\text{EXP}}$ is assigned to each damage class. Specifically, Softmax converts the real-valued vector $\boldsymbol{a}^{N_\ell} = (a_0^{N_\ell}, \ldots, a_{N_\ell}^{N_\ell})$*, provided

*It is worth noting that, in this case, the number of neurons in the last layer must match the number of target classes.

by the output layer of $\mathrm{NN}_{\mathrm{CL}}$, into the discrete probability distribution $\widehat{\mathbf{b}} = \mathrm{Softmax}(\boldsymbol{a}^{N_\ell})$, for $n = 0, \ldots, N_\ell$, as follows:

$$\widehat{b}^n(\boldsymbol{a}^{N_\ell}) = \frac{\exp(a_n^{N_\ell})}{\sum_{m=0}^{N_\ell} \exp(a_m^{N_\ell})}. \tag{4.9}$$

When the trained $\mathrm{NN}_{\mathrm{CL}}$ is employed for prediction, the best point estimate in $\widehat{\mathbf{b}} = \mathrm{NN}_{\mathrm{CL}}(\mathbf{U}^{\mathrm{EXP}})$ is selected as the class that best categorizes the processed measurements $\mathbf{U}^{\mathrm{EXP}}$. This setup characterizes the DL classifier for damage detection and localization used in Chapter 8 within the proposed DT framework.

4.2.3 Optimization for training deep models

The training or optimization of a DNN involves searching for the combination of its connection weights and biases that minimizes a loss function $\mathcal{L}$ on the training dataset $\mathcal{D}$. For ease of notation, the set of weights and biases is denoted as $\overline{\Xi}$, such that:

$$\overline{\Xi}^* = \arg\min_{\overline{\Xi}} \mathcal{L}(\mathcal{D}, \overline{\Xi}). \tag{4.10}$$

Typically, $\overline{\Xi}^*$ is sought iteratively using the backpropagation-based *gradient descent* training algorithm, which can be summarized in three steps:

I *Forward pass.* The training instance is processed to compute the output of each neuron, until the final output is obtained to evaluate the corresponding loss function score.

II *Backward pass.* The gradients of the loss function with respect to the learnable parameters are computed through the *backpropagation procedure* [244,245]. It begins by computing the gradient of the loss with respect to the output of each neuron in the output layer. These gradients are then back-propagated to the previous layers to compute the gradient of the loss with respect to the weighted sum of inputs for each neuron (by summing gradients from the layer above, each multiplied by the connection weight). Finally, the chain rule of derivation is applied to compute the gradients of the loss with respect to the learnable parameters. A step-by-step formalization can be found, e.g., in [103,251,252].

244. Rumelhart *et al.*, *Learning Internal Representations by Error Propagation*. 1987

245. Rumelhart *et al.*, *Learning representations by back-propagating errors*. 1986

III *Gradient descent.* The weights are updated using the gradient descent updating rule:

$$\overline{\Xi} \leftarrow \overline{\Xi} - \eta_{\overline{\Xi}} \nabla_{\overline{\Xi}} \mathcal{L}(\mathcal{D}, \overline{\Xi}), \tag{4.11}$$

where $\eta_{\overline{\Xi}} \in \mathbb{R}$ is the *learning rate* determining the size of the step taken during optimization, and $\nabla_{\overline{\Xi}} \mathcal{L}(\mathcal{D}, \overline{\Xi})$ are the gradients of the loss with respect to each parameter.

103. Goodfellow *et al.*, *Deep Learning*. 2016

251. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. 2019

252. Chollet, *Deep Learning with Python*. 2017

The learning rate $\eta_{\overline{\Xi}}$ controls the convergence speed of the training process. In practice, it is necessary to gradually decrease the learning rate as the training advances, allowing for large weight updates at the beginning of the training when $\overline{\Xi}$ is more distant from the optimum, and then progressively smaller corrections as the training proceeds.

Because gradient descent follows the direction where the gradient is maximal, it often displays an oscillatory pattern. This issue can be mitigated by introducing a *momentum term* $\mathbf{v}_{\overline{\Xi}}$ in the gradient descent updating rule:

$$\mathbf{v}_{\overline{\Xi}} \leftarrow \eta_{\mathbf{v}} \mathbf{v}_{\overline{\Xi}} - \eta_{\overline{\Xi}} \nabla_{\overline{\Xi}} \mathcal{L}(\mathcal{D}, \overline{\Xi}), \tag{4.12}$$

$$\overline{\Xi} \leftarrow \overline{\Xi} + \mathbf{v}_{\overline{\Xi}}, \tag{4.13}$$

where $\mathbf{v}_{\overline{\Xi}}$ has the same size as $\overline{\Xi}$ and can be interpreted as the direction and speed at which $\overline{\Xi}$ moves through the parameters space. The momentum is accumulated as an exponentially decaying moving average of past gradients. The hyperparameter $\eta_{\mathbf{v}} \in [0, 1)$ determines how quickly the contributions of previous gradients decay, and is typically adapted over time.

A common limitation of non-convex optimization problems is related to the selection of the starting location. At the beginning of the optimization, the learnable parameters are randomly initialized, following a *weight initialization scheme* chosen based on the employed activation function. The standard approach is to initialize the weights of DNNs equipped with "s"-shaped activation functions using the *Glorot's or Xavier's weight initialization* [253], and the weights of DNNs equipped with ReLU-like activation functions using *He's weight initialization* [254].

253. Glorot *et al.*, *Understanding the difficulty of training deep feedforward neural networks*. 2010

254. He *et al.*, *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. 2015

The gradients for updating $\overline{\Xi}$ can be computed by treating the dependence of $\mathcal{L}$ on $\mathcal{D}$ in three different ways:

I *Batch learning*. Batch methods process all the N_{train} training instances simultaneously in a single pass before computing the gradients for $\overline{\Xi}$.

II *Online learning*. Online methods process the training instances one by one, computing the gradients for updating $\overline{\Xi}$ individually for each instance. This method is typically employed when learning by drawing samples from a stream of training instances, rather than from a fixed-size dataset.

III *Mini-batch learning*. Mini-batch methods process the N_{train} training instances in sequences of small groups called *mini-batches*, and the gradients for updating $\overline{\Xi}$ are computed with reference to each mini-batch. These methods are typically referred to as stochastic methods. The typical example of

a stochastic method is *stochastic gradient descent*. The *mini-batch size* is a hyperparameter to be selected by considering several factors: (i) a larger mini-batch provides a more accurate estimate of the gradient; (ii) the training instances in a mini-batch are typically processed in parallel, and since the amount of required memory scales with the mini-batch size, hardware setups typically limit the mini-batch size; (iii) small mini-batch sizes can offer a regularizing effect due to the noise they add to the learning process; ($iiii$) the generalization error is often best for a mini-batch size of 1, i.e., online learning. However, training with such a small mini-batch size might require a small learning rate to maintain stability, and the total runtime can be very high. For these reasons, reducing the numbers of weight updates is typically preferred to decrease the computational effort and to have a more stable minimization.

The mini-batch learning mode is the standard choice for training NNs. During training, the N_{train} training instances in $\mathcal{D}$ are processed multiple times for updating $\overline{\Xi}$. Each time one pass of parameter update has been performed for all the N_{train} training instances, it is said that a training *epoch* has been completed.

The DL models employed in this Thesis are trained using *Adam* (adaptive moment estimation) [255], which is a first-order stochastic gradient descent optimizer that adaptively adjusts the learning rate for each learnable parameter individually. In particular, the learning rate is scaled by the moving average of the gradient and weighted by the square root of the moving average of the variance of the gradient. A bias correction is applied to the two moving averages to address initialization bias, enabling faster convergence during early training stages. Adam also takes advantage of momentum to stabilize the training. For a step-by-step formalization of stochastic descent optimization methods like Adam, the reader is referred to [103,252,251].

255. Kingma *et al.*, *Adam: A Method for Stochastic Optimization*. 2015

103. Goodfellow *et al.*, *Deep Learning*. 2016

252. Chollet, *Deep Learning with Python*. 2017

251. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. 2019

When the computational graph of a DNN becomes extremely deep, using the backpropagation method to update the learnable parameters via gradient descent implies the risk of *vanishing or exploding gradients* due to the repeated use of the chain rule of derivation through hidden layers. Vanishing gradients makes difficult to determine the direction in which the parameters should be updated, while exploding gradients can make learning unstable*. *Data normalization* through hidden layers is a family of techniques to mitigate gradient instability issues. These techniques scale the inputs or learnable parameters of a DNN in a way that makes the optimization more stable and efficient.

*The issue of vanishing or exploding gradients is particularly relevant for recurrent neural networks, as they use the same matrices of learnable parameters across each time step in temporal sequences, resulting in very deep computational graphs.

Batch normalization (BN) is a widely used technique for nor-

malizing the activations of DNNs layers [256]. It normalizes the activations for each mini-batch of inputs by subtracting the mean and dividing by the standard deviation of the activations for that mini-batch. During inference†, the statistics of the activations are obtained as a moving average of those observed during training. Additionally, scaling and offset factors are learned during training to restore the flexibility of the DNN after normalization is applied. We highlight that BN usually slows down the training process, as the normalization needs to be performed for each mini-batch.

256. Ioffe *et al.*, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*. 2015

†In the context of neural networks, "inference" refers to the evaluation of a trained model.

The design and training of DNN architectures requires choosing several parameters that are not learned during training but need to be selected in advance by the user. These are the so-called *hyperparameters*, which include the network topology, the regularization aspects, and the training options controlling the optimization algorithm. The performance of DL architectures critically depends on the specific choice of hyperparameters.

Throughout this Thesis, all choices concerning the DL architectures, as well as the relevant hyperparameters and training options, are made through a preliminary study. This study aims to minimize the objective loss function while retaining the generalization capabilities of the model, paying specific attention to avoiding overfitting of the training data. This involves monitoring *learning curves* that plot the evolution of the loss during training, as well as the generalization performance after training. Another viable option is to tune hyperparameters through an automated optimization scheme, such as random search [257], Bayesian optimization [258], and Hyperband [259].

257. Bergstra *et al.*, *Random Search for Hyper-Parameter Optimization*. 2012

258. Wu *et al.*, *Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization*. 2019

259. Li *et al.*, *Hyperband: A novel bandit-based approach to hyperparameter optimization*. 2017

4.2.4 Regularization for deep learning models

Regularization strategies are specifically designed to reduce the *generalization error*, possibly at the expense of an increased *training error*. In overfitting regime, the variance rather than bias dominates the prediction error. Regularization strategies work by trading a slightly increased bias for a significantly reduced variance. In this Section, we present several regularization procedures that are employed in this Thesis to prevent overfitting, namely: *parameter norm penalty*, *early stopping*, *dataset augmentation*, *dropout*, and *skip connections*.

Parameter norm penalty. These regularization approaches involve constraining the NN *representation capabilities* by augmenting the loss function with a norm penalty on the learnable parameters. The norm penalty is typically applied only to the weights, leaving the biases unregularized, as these typically require less data than the weights to fit accurately.

Different norms have varying effects on the regularization behavior. The L^2 parameter norm penalty, known as *weight decay* or *ridge regression* or *Tikhonov regularization*, constraints the $N_\omega \in \mathbb{N}$ connection weights collected in Ξ, by adding to the loss function a regularization term $\lambda_{L^2}\|\mathtt{vec}[\Xi]\|_2^2$, to provide the following *regularized loss function*:

$$\begin{aligned}\mathcal{L}^{R2}(\mathcal{D},\overline{\Xi}) &= \mathcal{L}(\mathcal{D},\overline{\Xi}) + \lambda_{L^2}\|\mathtt{vec}[\Xi]\|_2^2 = \\ &= \mathcal{L}(\mathcal{D},\overline{\Xi}) + \lambda_{L^2}\sum_{n_\omega}^{N_\omega}\omega_{n_\omega}^2.\end{aligned} \tag{4.14}$$

In Eq. (4.14): $\mathcal{L}$ is the unregularized loss function and $\overline{\Xi}$ is the full set of weights and bias parametrizing the DNN, with $\overline{\Xi} \supseteq \Xi$; $\lambda_{L^2} \in \mathbb{R}$ is the L^2 regularization rate applied over Ξ; and $\mathtt{vec} : \mathbb{R}^{N_R \times N_C} \longrightarrow \mathbb{R}^{N_R N_C}$ denotes the *vectorization operation* that converts a matrix $\mathbf{A} \in \mathbb{R}^{N_R \times N_C}$ into a column vector by stacking its columns on top of one another. The gradient of $\mathcal{L}^{R2}$ with respect to the generic n_ω-th learnable weight ω_{n_ω}, with $n_\omega = 1, \ldots, N_\omega$, is given by:

$$\frac{\partial \mathcal{L}^{R2}}{\partial \omega_{n_\omega}} = \frac{\partial \mathcal{L}}{\partial \omega_{n_\omega}} + 2\lambda_{L^2}\omega_{n_\omega}. \tag{4.15}$$

The gradient descent rule to update ω_{n_ω} thus becomes:

$$\begin{aligned}\omega_{n_\omega} \leftarrow &\omega_{n_\omega} - \eta_{\overline{\Xi}}\frac{\partial \mathcal{L}}{\partial \omega_{n_\omega}} - 2\eta_{\overline{\Xi}}\lambda_{L^2}\omega_{n_\omega} = \\ =&\omega_{n_\omega}(1 - 2\eta_{\overline{\Xi}}\lambda_{L^2}) - \eta_{\overline{\Xi}}\frac{\partial \mathcal{L}}{\partial \omega_{n_\omega}}.\end{aligned} \tag{4.16}$$

Eq. (4.16) shows that the regularization term influences the learning by scaling the weights of the model by a constant factor before they are updated in the gradient direction. Only directions along which the parameters significantly contribute to reducing the loss function are preserved relatively intact. A large λ_{L^2} value forces ω_{n_ω} to be close to zero, consequently reducing the representation capabilities of the model. In contrast, a small λ_{L^2} allows the model to easily fit the training data. This shrinking effect implies that the model behavior remains stable for similar inputs, preventing it from learning local noise in the data. The L^2 parameter norm penalty can be seen as defining a Gaussian prior for ω_{n_ω} of the form $\omega_{n_\omega} \sim \mathcal{N}(0, \frac{1}{\lambda_{L^2}})$, whose optimization involves finding the corresponding maximum a-posteriori estimate.

Early stopping. As the number of training epochs increases, the loss function for the training dataset ideally decreases, indicating an improved fitting ability of the NN. However, if we

continue optimizing the learnable parameters of an NN with a sufficient *representational capacity*, there reaches a point where the loss function score on the training set keeps decreasing, but the score on the validation set starts to rise. This means that the model has begun *overfitting the training data*. An effective way to interrupt the learning process before overfitting occurs is to monitor the loss function evolution at the end of each training epoch on a validation set. Each time the error on the validation set improves, a copy of the learnable parameters is stored. Training is then interrupted whenever the loss function score on the validation set has not decreased for a prescribed number of consecutive "patience" epochs. By retrieving the setting of learnable parameters associated with the minimum validation loss score, this approach allows obtaining a model with a better validation error, and potentially, a better generalization error.

Dataset augmentation. The most effective way to enhance the generalization capabilities of an NN is to train it on a larger dataset. A convenient approach to access additional data is by generating synthetic data and incorporating them into the training set. This allows the NN to be invariant to a range of transformations applied to generate the new data. This strategy is particularly advantageous for high-dimensional data such as time series and images, due to the large range of transformations that can be simulated. In general, depending on the task at hand, hand-designed augmentation schemes can significantly reduce the generalization error of an NN. *Noise injection* is a form of dataset augmentation that we systematically employ in this Thesis to enhance the robustness of NNs to measurement noise*. This involves training NNs with noisy inputs, as proposed in [260]. In our experiments, we model this corruption as an *additive, independent, identically distributed Gaussian noise* drawn from a known noise model. This allows for mimicking signal perturbations affecting real-life sensor measurements, including those from micro electro-mechanical accelerometers [261,262].

*In practice, it is often the case that even if the asset is perfectly modeled by physics-based models, the data we actually observe are prone to corruption, such as sensor noise or inspection errors.

260. Vincent *et al.*, *Extracting and composing robust features with denoising autoencoders*. 2008

261. D'Alessandro *et al.*, *Characterization of MEMS accelerometer self-noise by means of PSD and Allan Variance analysis*. 2017

262. Evans *et al.*, *Performance of Several Low-Cost Accelerometers*. 2014

Dropout. Dropout (Dpout) can be considered as a method to make *bagging* computationally inexpensive for exponentially large ensembles of NNs [263]. Bagging involves training and evaluating multiple models. However, when dealing with large NNs as individual models, this approach becomes impractical due to computational constraints. The Dpout approximation to bagging is achieved by training an ensemble of NNs, consisting of sub-networks formed by *masking* non-output neurons from the underlying base NN. Each time a training instance is loaded into a mini-batch, a different binary mask is randomly sampled and applied to the regularized units. The mask for each unit is

263. Srivastava *et al.*, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. 2014

sampled independently from all the others, and the probability of sampling a masked value is a hyperparameter. Only a fraction of all the possible sub-networks undergoes training, and the parameter sharing causes the remaining sub-networks to arrive at good settings of the learnable parameters. Each hidden unit must perform well regardless of the presence or absence of other hidden units in the model. In this light, Dpout regularizes each hidden unit to provide not merely a good feature, but a feature that is good in many contexts.

Skip connections. Originally developed within the computer vision community [264], skip connections, or *residual connections*, help to avoid vanishing/exploding gradients and mitigate the *accuracy saturation* problem[†][265,266,267]. A *residual block* exploits an identity mapping of the input to provide an additional path towards the terminal part of the network by skipping some layers. This extra path is then added to the outputs of the stacked layers through an element-wise summation. The underlying idea is that it is easier for the learnable layers to fit the residual of the desired mapping rather than the desired mapping itself. Let $\widetilde{\mathcal{F}}(\boldsymbol{a})$ be the desired mapping and $\mathcal{F}(\boldsymbol{a})$ a learnable mapping of $\boldsymbol{a}$. Instead of learning $\widetilde{\mathcal{F}}(\boldsymbol{a})$ directly, skip connections allow fitting $\mathcal{F}(\boldsymbol{a})$ to the residual mapping $\widetilde{\mathcal{F}}(\boldsymbol{a}) - \boldsymbol{a}$. An approximation to the desired mapping is then retrieved as $\widetilde{\mathcal{F}}(\boldsymbol{a}) \approx \mathcal{F}(\boldsymbol{a}) + \boldsymbol{a}$. Multiple residual blocks are usually exploited within the same architecture to perform an iterative refinement of the output features, with each block slightly improving the representation of the previous layer.

264. He *et al.*, *Deep Residual Learning for Image Recognition*

[†]The "accuracy saturation" problem implies that both the training and generalization errors do not improve as the deep neural network gets deeper.

265. Jastrzębski *et al.*, *Residual Connections Encourage Iterative Inference*. 2018

266. Bello *et al.*, *Revisiting ResNets: Improved Training and Scaling Strategies*. 2021

267. Veit *et al.*, *Residual networks behave like ensembles of relatively shallow networks*. 2016

4.2.5 1D convolutional layer

CNNs draw inspiration from the concepts of simple and complex cells in visual neuroscience, embodying an architecture reminiscent of the hierarchy found in the visual cortex of living organisms. Pioneering research in the late 1950s [268] revealed that visual neurons at higher levels establish connections solely with neurons within their *receptive field* at lower levels. Consequently, each neuron at a higher level is envisioned to be connected only to few selected neurons from the preceding level. Moreover, the receptive fields of individual neurons can overlap, collectively shaping the *visual field*. The CNN architecture aims to replicate this pattern and is built upon the repetition of blocks uncommon to the multi-layer perceptron architecture.

Initially developed within the computer vision community [269,270,271], CNNs have rapidly become the first choice for addressing various problems, outperforming alternative ML

268. Hubel, *Single unit activity in striate cortex of unrestrained cat*. 1959

269. LeCun *et al.*, *Handwritten Digit Recognition with a Back-Propagation Network*. 1989

270. Krizhevsky *et al.*, *ImageNet Classification with Deep Convolutional Neural Networks*. 2012

271. LeCun *et al.*, *Single unit activity in striate cortex of unrestrained cats*. 1959

methods. Their success can be attributed to: (i) the synergy of the feature selection and feature extraction stages into a single learning body; (ii) their *sparse connectivity* and *parameter sharing*, leading to significant computational efficiency compared to fully-connected NNs; (iii) their *tolerance for translation, scaling, skewing, and distortion* of the input data.

CNNs are designed to process data in the form of multiple arrays — one-dimensional (1D) for signals and sequences, two-dimensional (2D) for images or spectrograms, and three-dimensional (3D) for video or volumetric images [272]. CNNs exploit the characteristic of data as *compositional hierarchies*, where higher-level features are formed by composing lower-level ones. Structured as a series of modules, each typically comprising a *convolutional layer* and a *pooling layer*, CNNs organize computational units in *feature maps*. In a convolutional layer, each unit connects to local patches in the feature maps of the previous layer through learnable weights called *convolutional filters* or *filter kernels*. These units are not connected to all inputs in the previous layer, but only to those within their receptive field. Multiple filters are applied to the inputs of each convolutional layer, resulting in multiple feature maps. The receptive field of an individual neuron is active across all feature maps from the previous layer. Each feature map shares the same filters, but different feature maps in a layer use different filters — if a pattern can be detected in one part of the input, it could be so anywhere. The filtering operation performed by a feature map is a *discrete convolution*, and convolutional layers are NN layers that use convolution in place of matrix multiplication.

272. Avci *et al.*, *A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications*. 2021

For ease of notation, we now discuss 1D CNNs, focusing on their application to multivariate time series. However, the same principles can be similarly extended to both 2D and 3D convolutional layers. The computational model of the 1D convolutional layer is presented with reference to a generic ℓ-th convolutional layer $C_\ell : \mathbb{R}^{L_\ell \times N_F^\ell} \mapsto \mathbb{R}^{L_\ell \times N_F^{\ell+1}}$, $\boldsymbol{\mathcal{A}}^\ell \mapsto \widetilde{\boldsymbol{\mathcal{A}}}^\ell$, where $\ell = 1, \ldots, N_\ell$, in a DNN comprising N_ℓ layers. Denoting the input tensor as $\boldsymbol{\mathcal{A}}^\ell = [\boldsymbol{a}_1^\ell, \ldots, \boldsymbol{a}_{N_F^\ell}^\ell]$, composed of $N_F^\ell \in \mathbb{N}$ arrays of length $L_\ell \in \mathbb{N}$, C_ℓ provides a set of $N_F^{\ell+1} \in \mathbb{N}$ output feature maps denoted as $\widetilde{\boldsymbol{\mathcal{A}}}^\ell = [\widetilde{\boldsymbol{a}}_1^\ell, \ldots, \widetilde{\boldsymbol{a}}_{N_F^{\ell+1}}^\ell]$, computed as:

$$\widetilde{\boldsymbol{a}}_{n_{F*}}^\ell = C_\ell(\boldsymbol{\mathcal{A}}^\ell) = \mathcal{A}(\sum_{n_F=1}^{N_F^\ell} \boldsymbol{\Theta}_{n_{F*}}^{\ell n_F} * \boldsymbol{a}_{n_F}^\ell + \beta_{n_{F*}}^\ell), \tag{4.17}$$

with $n_{F*} = 1, \ldots, N_F^{\ell+1}$. In Eq. (4.17), the discrete convolution operator is denoted as $* : (\mathbb{R}^{H_C^\ell} \times \mathbb{R}^{L_\ell}) \mapsto \mathbb{R}^{L_\ell}$, which performs cross-

correlation with flipped kernel. The 1D discrete convolution operator works by *sliding* the filter kernels along the data dimension (the time dimension in signal processing). Here, $H_C^\ell \in \mathbb{N}$ is the size of the filter kernels, determining the dimension of the local correlation looked at by the convolutional layer [273]. A larger kernel size implies a larger receptive field but also involves tuning more parameters. However, the receptive field can be conveniently expanded by stacking more convolutional layers. The n_F-th filter kernel that the n_{F*}-th feature map applies to the input tensor is denoted as $\boldsymbol{\Theta}_{n_{F*}}^{\ell\, n_F} \in \mathbb{R}^{H_C^\ell}$. The set of filter kernels that the n_{F*}-th feature map simultaneously applies to the input tensor is denoted as $\boldsymbol{\Theta}_{n_{F*}}^{\ell} = \{\boldsymbol{\Theta}_{n_{F*}}^{\ell\, n_F}\}_{n_F=1}^{N_F^\ell} \in \mathbb{R}^{N_F^\ell \times H_C^\ell}$, with $n_F = 1, \ldots, N_F^\ell$ and $n_{F*} = 1, \ldots, N_F^{\ell+1}$. Finally, $\boldsymbol{\Theta}^\ell = \{\boldsymbol{\Theta}_{n_{F*}}^{\ell}\}_{n_{F*}=1}^{N_F^{\ell+1}} \in \mathbb{R}^{N_F^{\ell+1} \times N_F^\ell \times H_C^\ell}$ represents the set of N_F learnable filters for the ℓ-th convolutional layer. Additionally, $\beta_{n_{F*}}^{\ell} \in \mathbb{R}$ and $\mathcal{A} : \mathbb{R} \mapsto \mathbb{R}$ represent the bias term for the n_{F*}-th feature map and a nonlinear activation function, respectively. The length L_ℓ of the output feature maps can be reduced by skipping with a certain integer step, termed *stride*, some entries between two adjacent receptive fields along the sliding direction. To maintain consistency in the L_ℓ dimension between $\boldsymbol{\mathcal{A}}^\ell$ and $\widetilde{\boldsymbol{\mathcal{A}}}^\ell$, *zero padding* is typically used to extend with zeros the limits of $\boldsymbol{\mathcal{A}}^\ell$. If 1D convolutional layers are employed in building the NN architecture, the resulting NN architecture is defined as a 1D CNN.

273. Hu *et al.*, *Squeeze–and–Excitation Networks*. 2018

Thanks to their sparse connectivity, the number of learnable parameters in convolutional layers can be kept consistently small compared to fully-connected layers. Moreover, convolutional layers inherently embody valuable relational *inductive biases*, such as *locality* and *translation equivariance** [103,274]. These learning biases are particularly effective for detecting time correlations within and across time series [275], a crucial aspect for SHM purposes. Thanks to their ability to automatically select and extract damage-sensitive features from raw structural vibrations, CNNs have recently showcased state-of-the-art performance in vibration-based SHM of civil structures [95,276,277,278].

*An "inductive bias", or learning bias, is an assumption embedded in the model, that it exploits to predict the output. "Locality" refers to the proximity of convolutional kernel inputs and their isolation from distant entities. "Translational equivariance" refers to the ability to apply the learned rule across different localities in the input. This ensures that if the input changes, the output changes in the same way.

103. Goodfellow *et al.*, *Deep Learning*. 2016

274. Battaglia *et al.*, *Relational inductive biases, deep learning, and graph networks*. 2018

275. Wang *et al.*, *Time series classification from scratch with deep neural networks: A strong baseline*. 2017

95. Abdeljaber *et al.*, *Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks*. 2017

276. Zhang *et al.*, *SHMnet: Condition assessment of bolted connection with beyond human-level performance*. 2020

277. Ewald *et al.*, *DeepSHM: a deep learning approach for structural health monitoring based on guided Lamb wave technique*. 2019

278. Giglioni *et al.*, *Autoencoders for unsupervised real-time bridge health assessment*. 2022

The convolutional layers capability of enabling meaningful information detected in a certain portion of the input to occupy a specific position in the output is not always desirable. It may be preferable to consolidate meaningful information over the data dimension L_ℓ. In the context of CNNs, a *pooling layer* synthesizes $\widetilde{\boldsymbol{\mathcal{A}}}^\ell$ by down-sampling neighbouring shifted patches, thereby establishing invariance to shifts and distortions. Various types of pooling layer exist. One example is 1D *max pooling* [279], denoted as $\mathcal{P}_{\max}^\ell : \mathbb{R}^{L_\ell \times N_F^{\ell+1}} \mapsto \mathbb{R}^{L_{\ell+1} \times N_F^{\ell+1}}$, with

279. Zhou *et al.*, *Computation of optical flow using a neural network*. 1988

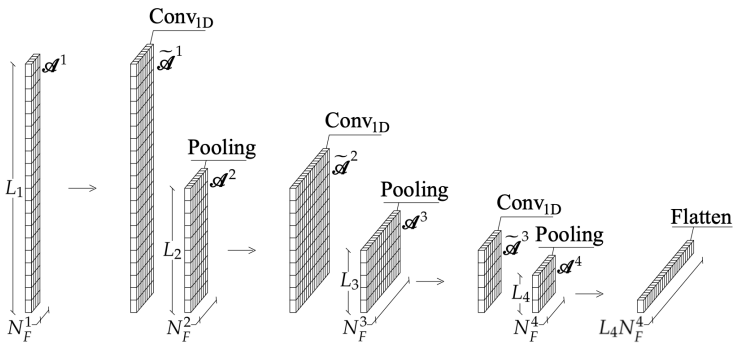


Figure 4.2: Exemplary convolutional neural network architecture featuring three convolutional modules, each composed of a one-dimensional convolutional layer and a pooling layer. The output is then reshaped through a flatten layer.

$\ell = 1, \dots, N_\ell$, which provides $\mathscr{A}^{\ell+1} = \mathcal{P}^\ell_{\max}(\widetilde{\mathscr{A}}^\ell)$ by retaining only the maximum value in each data pool. Depending on the chosen *pool size* $H^\ell_{\mathcal{P}} \in \mathbb{N}$ and the *stride width* $\Xi^\ell_{\mathcal{P}} \in \mathbb{N}$, the resulting output size is given by $L_{\ell+1} = (L_\ell - H^\ell_{\mathcal{P}} + 1)/\Xi^\ell_{\mathcal{P}}$.

The typical CNN architecture is obtained by alternating convolutional and pooling layers that increase the number of feature maps while simultaneously reducing input dimensions. The output of the last pooling layer is usually *flattened* and then connected to a fully-connected layer. See Fig. 4.2 for a simplified scheme of a CNN. CNNs are extensively exploited in this Thesis, mainly because of their ability to automatically select and extract damage-sensitive features from raw data.

4.2.6 Long Short-Term Memory Layer

RNNs are a family of NNs designed for processing sequential data. They can handle variable-length *sequences* of inputs by leveraging the *time ordering* of the information flow and processing one element in the input sequence at a time. This capability is enabled by *hidden units* that contain a *hidden state variable*, acting as a *data memory* and capturing *time dependencies* [103].

Typically, RNNs are more suitable than fully-connected ones for time series and time-dependent problems [150,280,281,282]. A related idea is the use of convolution across a 1D temporal sequence. However, RNNs share parameters in a different way than 1D CNNs. Indeed, the RNN output at each time step is a function of the current input and the hidden state from the previous time step, and it is computed using the same weights and biases consistently over time.

Once *unrolled* in time, RNNs can be seen as very deep NNs in which all the layers share the same weights. Despite their primary purpose is to learn *long-term dependencies*, it is often challenging to train them to store information for extended periods due to vanishing gradients [103]. Specifically, the gradient of a long-term interaction has an exponentially smaller magnitude

103. Goodfellow *et al.*, *Deep Learning*. 2016

150. Conti *et al.*, *Multi-fidelity surrogate modeling using long short-term memory networks*. 2023

280. Zhang *et al.*, *Physics-informed multi-LSTM networks for metamodeling of nonlinear structures*. 2020

281. Zhang *et al.*, *Deep long short-term memory networks for nonlinear structural seismic response prediction*. 2019

282. Simpson *et al.*, *Machine Learning Approach to Model Order Reduction of Nonlinear Systems via Autoencoder and LSTM Networks*. 2021

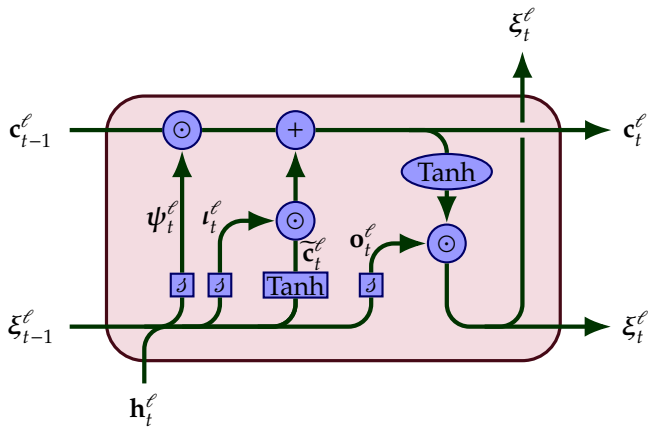


Figure 4.3: Architecture of the long short-term memory cell. (figure adapted from [284])

than the gradient of a short-term interaction. To overcome this limitation, the long short-term memory (LSTM) cell model [283] is a RNN designed to better capture long-term dependencies through an internal *gating mechanism* that recognizes important components in the information flow [103].

283. Hochreiter *et al.*, *Long short-term memory*. 1997

In the following, we present the standard LSTM cell, as proposed in [283] and illustrated in Fig. 4.3. At the core of the model there is the *internal cell*, which retains the cell state from the preceding time step through a *self-recurrent connection* where the gradient can flow for an extended duration. The cell state gets updated through three interconnected gates, designed to incorporate or discard information depending on the context: the *forget gate*, the *input gate*, and the *output gate*. The comprehensive gating mechanism comprises four units, each resembling a fully-connected layer and depicted as blue boxes in Fig. 4.3. These units are used at every time step to decide when to clear the memory content. Here, $\mathcal{S}$ and Tanh denote sigmoid-activated and Tanh-activated fully-connected layers, respectively.

In the following, we discuss the LSTM cell with reference to a generic ℓ-th LSTM layer $\mathcal{R}^\ell : (\mathbb{R}^{N_c^\ell}, \mathbb{R}^{N_c^\ell}, \mathbb{R}^{N_h^\ell}) \mapsto (\mathbb{R}^{N_c^\ell}, \mathbb{R}^{N_c^\ell})$, $(\xi_{t-1}^\ell, \mathbf{c}_{t-1}^\ell, \mathbf{h}_t^\ell) \mapsto (\xi_t^\ell, \mathbf{c}_t^\ell)$, with $\ell = 1, \ldots, N_\ell$, in a DNN comprising N_ℓ layers. The overall LSTM cell adheres to the following computational model:

$$\psi_t^\ell = \mathcal{S}(\mathbf{\Omega}_{h\psi}^\ell \mathbf{h}_t^\ell + \mathbf{\Omega}_{\xi\psi}^\ell \xi_{t-1}^\ell + \boldsymbol{\beta}_\psi^\ell), \tag{4.18}$$

$$\iota_t^\ell = \mathcal{S}(\mathbf{\Omega}_{h\iota}^\ell \mathbf{h}_t^\ell + \mathbf{\Omega}_{\xi\iota}^\ell \xi_{t-1}^\ell + \boldsymbol{\beta}_\iota^\ell), \tag{4.19}$$

$$\widetilde{\mathbf{c}}_t^\ell = \mathrm{Tanh}(\mathbf{\Omega}_{hc}^\ell \mathbf{h}_t^\ell + \mathbf{\Omega}_{\xi c}^\ell \xi_{t-1}^\ell + \boldsymbol{\beta}_c^\ell), \tag{4.20}$$

$$\mathbf{c}_t^\ell = \psi_t^\ell \odot \mathbf{c}_{t-1}^\ell + \iota_t^\ell \odot \widetilde{\mathbf{c}}_t^\ell, \tag{4.21}$$

$$\mathbf{o}_t^\ell = \mathcal{S}(\mathbf{\Omega}_{ho}^\ell \mathbf{h}_t^\ell + \mathbf{\Omega}_{\xi o}^\ell \xi_{t-1}^\ell + \boldsymbol{\beta}_o^\ell), \tag{4.22}$$

$$\xi_t^\ell = \mathbf{o}_t^\ell \odot \mathrm{Tanh}(\mathbf{c}_t^\ell). \tag{4.23}$$

Here, $\mathbf{c}_{t-1}^{\ell} \in \mathbb{R}^{N_c^{\ell}}$ represents the *cell state* at the previous time step $t-1$, with size $N_c^{\ell} \in \mathbb{N}$. $\mathbf{h}_t^{\ell} \in \mathbb{R}^{N_h^{\ell}}$ denotes the *input features* at the current time step t, with size $N_h^{\ell} \in \mathbb{N}$. The *hidden state* at time $t-1$ is denoted as $\boldsymbol{\xi}_{t-1}^{\ell} \in \mathbb{R}^{N_c^{\ell}}$. The weights matrices of the four dense layers in the LSTM cell are denoted as $\boldsymbol{\Omega}_{h\square}^{\ell} \in \mathbb{R}^{N_c^{\ell} \times N_h^{\ell}}$ and $\boldsymbol{\Omega}_{\xi\square}^{\ell} \in \mathbb{R}^{N_c^{\ell} \times N_c^{\ell}}$, respectively applied to $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$, for $\square \in \{\psi, \iota, c, o\}$. The bias vectors are denoted as $\boldsymbol{\beta}_{\square}^{\ell} \in \mathbb{R}^{N_c^{\ell}}$.

The ℓ-th *forget gate* takes $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$, and run them through a sigmoid-activated layer to output a vector $\boldsymbol{\psi}_t^{\ell} \in \mathbb{R}^{N_c^{\ell}}$ with values between 0 and 1, according to Eq. (4.18). The two extreme values correspond to either completely keep or completely get rid of the current cell state.

The ℓ-th *input gate* also takes as input $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$, but it regulates the amount of new information to let through the new cell state. First, a sigmoid-activated layer takes $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$ to detect the values to be updated in the cell state, as provided by $\boldsymbol{\iota}_t^{\ell} \in \mathbb{R}^{N_c^{\ell}}$ in Eq. (4.19). Then, a Tanh-activated layer takes $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$, and computes the new candidate values $\widetilde{\mathbf{c}}_t^{\ell} \in \mathbb{R}^{N_c^{\ell}}$ using Eq. (4.20). Finally, a *new cell state* $\mathbf{c}_t^{\ell}$ is computed by summing $\mathbf{c}_{t-1}^{\ell}$ and $\widetilde{\mathbf{c}}_t^{\ell}$, respectively scaled by $\boldsymbol{\psi}_t$ and $\boldsymbol{\iota}_t^{\ell}$, see Eq. (4.21).

The ℓ-th *output gate* takes as input $\mathbf{h}_t^{\ell}$, $\boldsymbol{\xi}_{t-1}^{\ell}$ and $\mathbf{c}_t^{\ell}$ to update the hidden state $\boldsymbol{\xi}_t^{\ell}$ for the current time step. First, a sigmoid-activated layer takes $\mathbf{h}_t^{\ell}$ and $\boldsymbol{\xi}_{t-1}^{\ell}$ to provide $\boldsymbol{o}_t^{\ell} \in \mathbb{R}^{N_c^{\ell}}$, which selects the information in the cell state to be taken into account, see Eq. (4.22). Finally, the *updated hidden state* $\boldsymbol{\xi}_t^{\ell}$ is computed by taking the Tanh of the current cell state $\mathbf{c}_{t-1}^{\ell}$, and then gating the resulting activation via $\boldsymbol{o}_t^{\ell}$, according to Eq. (4.23).

In a RNN, gradients are backpropagated through the unrolled computational graph. However, because hidden units in RNNs depend on previous time steps, a *backpropagation through time* algorithm is necessary to compute the gradients. To mitigate the computational burden of backpropagating through time, the input sequences can be *chunked* into (potentially overlapping) smaller sequences treated as independent training samples. The chunks length is a hyperparameter to be chosen for optimal computational performance.

In this Thesis, the LSTM model is employed in Chapter 6 within the proposed MF-DNN surrogate modeling strategy to leverage the *time correlation* between signals of different fidelities and adaptively refine the LF signals with the effect of damage.

4.2.7 AUTOENCODERS

An AE is a popular DNN architecture that is typically trained to replicate its input at its output. It can be conceptualized as com-

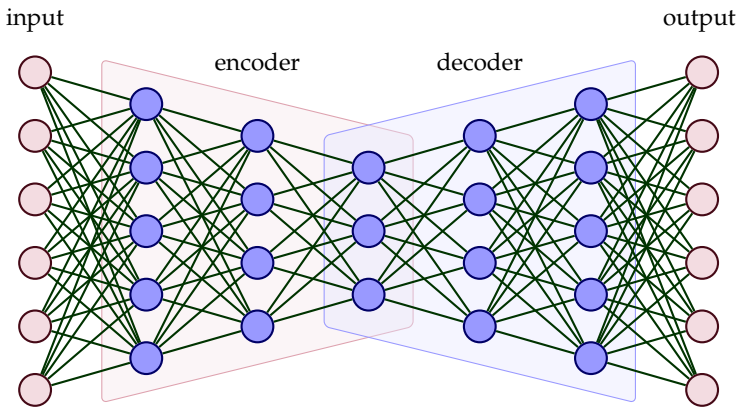


Figure 4.4: Exemplary autoencoder architecture featuring five fully-connected hidden layers: red nodes denote the input/output quantities, while blue nodes are the learnable neurons.

prising two components: an *encoder branch* and a *decoder branch*. Typically, it includes a hidden *bottleneck layer* featuring a smaller dimension than the input and output layers. This configuration forces the data through a *low-dimensional representation* that must capture the most salient features of the data (see Fig. 4.4).

Traditionally, AEs have been used for tasks such as dimensionality reduction or feature learning [285,286,287,288,289,290,291], as well as *generative modeling* [292,293,294,295,296,297,298]. Relevant applications of AEs in the field of SHM include data normalization [108,299], feature extraction [300,301], anomaly detection [278,302,303], as well as surrogate modeling and model discovery [282,304,305,306], among the others.

The dataset used for training an AE follows the *unsupervised learning* setting, where each sample serves as both the input and the output variable. For our SHM purposes, the AE architecture can be employed for dimensionality reduction of high-dimensional vibration recordings. In this context, the relevant dataset is represented as:

$$\mathcal{D}_{\text{AE}} = \{\mathbf{U}_i\}_{i=1}^{N_{\text{train}}}, \tag{4.24}$$

where synthetic vibration recordings $\mathbf{U}_i$ are generated by numerically simulating the structural response for the i-th sampling of $\boldsymbol{\mu}_i$, with $i = 1, \ldots, N_{\text{train}}$, following the procedure detailed in Sec. 3.3.2. The dimensionality reduction process underlying the *latent representation** is only driven by the factors of variation characterizing the training data.

The learnable components of an AE are its encoding and decoding branches, denoted as NN_{ENC} and NN_{DEC}, respectively. The encoding branch, $\text{NN}_{\text{ENC}} : \mathbb{R}^{L \times N_u} \mapsto \mathbb{R}^{N_h}$, $\mathbf{U} \mapsto \mathbf{h}$, provides the feature representation $\mathbf{h}$ of the input vibration recordings $\mathbf{U}(\boldsymbol{\mu})$ in a low-dimensional space of size N_h. On the other hand, the decoding branch, $\text{NN}_{\text{DEC}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}^{L \times N_u}$, $\mathbf{h} \mapsto \mathbf{U}$, takes

285. Kramer, *Nonlinear principal component analysis using autoassociative neural networks*. 1991
286. Kerschen *et al.*, *Feature extraction using auto-associative neural networks*. 2003
287. Hinton *et al.*, *Reducing the Dimensionality of Data with Neural Networks*. 2006
288. Hinton *et al.*, *Transforming auto-encoders*. 2011
289. Vincent *et al.*, *Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion*. 2010
290. Liou *et al.*, *Autoencoder for words*. 2014
291. Japkowicz *et al.*, *Nonlinear autoassociation is not equivalent to PCA*. 2000
292. Kingma *et al.*, *An Introduction to Variational Autoencoders*. 2019
293. Salakhutdinov *et al.*, *Semantic hashing*. 2009
294. Kingma *et al.*, *Auto-Encoding Variational Bayes*. 2013
295. Makhzani *et al.*, *Adversarial Autoencoders*. 2015
296. Tolstikhin *et al.*, *Wasserstein Auto-Encoders*. 2017
297. Sohn *et al.*, *Learning Structured Output Representation using Deep Conditional Generative Models*. 2015
298. Higgins *et al.*, *beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework*. 2017
108. Sohn *et al.*, *Statistical Damage Classification Under Changing Environmental and Operational Conditions*. 2002

$\mathbf{h}(\mathbf{U})$ and provides the reconstructed input $\widehat{\mathbf{U}}(\mathbf{h})$, as follows:

$$\mathbf{h}(\mathbf{U}) = \mathrm{NN}_{\mathrm{ENC}}(\mathbf{U}(\boldsymbol{\mu}))\ , \tag{4.25}$$

$$\widehat{\mathbf{U}}(\mathbf{h}) = \mathrm{NN}_{\mathrm{DEC}}(\mathbf{h}(\mathbf{U}))\ . \tag{4.26}$$

The key component connecting $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{DEC}}$ is the bottleneck layer, characterized by the low-dimensional feature size N_h. This much smaller dimension than that of the input and output layers compresses the data representation while aiming to recreate the input at the output.

The learnable parameters of $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{DEC}}$, denoted as $\boldsymbol{\Omega}_{\mathrm{AE}}$, are optimized to statistically model the *input-output relationship* underlying $\mathcal{D}_{\mathrm{AE}}$. This optimization is achieved by minimizing the following loss function over $\mathcal{D}_{\mathrm{AE}}$:

$$\begin{aligned}\mathcal{L}_{\mathrm{AE}}(\boldsymbol{\Omega}_{\mathrm{AE}}, \mathcal{D}_{\mathrm{AE}}) = \\ \frac{1}{N_{\mathrm{train}}} \sum_{i=1}^{N_{\mathrm{train}}} \|\mathtt{vec}[\mathbf{U}_i(\boldsymbol{\mu}_i) - \mathrm{NN}_{\mathrm{DEC}}(\mathrm{NN}_{\mathrm{ENC}}(\mathbf{U}_i(\boldsymbol{\mu}_i)))]\|_2^2 + \\ \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{AE}}]\|_2^2.\end{aligned} \tag{4.27}$$

In Eq. (4.27), the first term is an MSE-based reconstruction loss function commonly used for training AEs in a regression setting. The second term is an L^2 regularization of rate λ_{L^2}, applied over the learnable parameters $\boldsymbol{\Omega}_{\mathrm{AE}}$. This loss function represents a balance between two objectives. First, it aims to learn a representation $\mathbf{h}$ for any given $\mathbf{U}$ in $\mathcal{D}_{\mathrm{AE}}$, allowing to recover $\mathbf{U}$ through $\mathrm{NN}_{\mathrm{DEC}}$. Second, the AE needs to satisfy the norm penalty imposed over $\boldsymbol{\Omega}_{\mathrm{AE}}$ to limit its representation capabilities. Satisfying this compromise encourages the low-dimensional representation $\mathbf{h}$ to capture the *data-generating distribution* underlying $\mathcal{D}_{\mathrm{AE}}$. If the data-generating distribution concentrates near a *low-dimensional manifold*, the encoded representations implicitly capture a coordinate system for this manifold.

It is worth noting that, thanks to the *nonlinear dimensionality reduction* capabilities of AEs, it is unnecessary to pre-process $\mathcal{D}_{\mathrm{AE}}$ for feature extraction purposes, as the AE is designed to accomplish this process. However, in the unsupervised setting, there is no guarantee that the extracted features are representative of structural health parameters. This situation stands in contrast with that characterizing the supervised training of a DL-based regressor or classifier, where the model learn to extract a feature representation optimized for a specific task. Therefore, pre-processing $\mathcal{D}_{\mathrm{AE}}$ to highlight damage-sensitive patterns could be beneficial in biasing the dimensionality reduction process toward the envisioned downstream task. An

299. Sohn *et al.*, *Novelty detection under changing environmental conditions*. 2001

300. Rosafalco *et al.*, *An Autoencoder-Based Deep Learning Approach for Load Identification in Structural Dynamics*. 2021

301. Ma *et al.*, *Structural damage identification based on unsupervised feature-extraction via Variational Auto-encoder*. 2020

278. Giglioni *et al.*, *Autoencoders for unsupervised real-time bridge health assessment*. 2022

302. Worden *et al.*, *Experimental validation of a structural health monitoring methodology: Part I. Novelty detection on a laboratory structure*. 2003

303. Pathirage *et al.*, *Structural damage identification based on autoencoder neural networks and deep learning*. 2018

282. Simpson *et al.*, *Machine Learning Approach to Model Order Reduction of Nonlinear Systems via Autoencoder and LSTM Networks*. 2021

304. Simpson *et al.*, *Reduced order modeling of non-linear monopile dynamics via an AE-LSTM scheme*. 2023

305. Liu *et al.*, *Physics-guided Deep Markov Models for learning nonlinear dynamical systems with uncertainty*. 2022

306. Mylonas *et al.*, *Conditional variational autoencoders for probabilistic wind turbine blade fatigue estimation using Supervisory, Control, and Data Acquisition data*. 2021

*The term "latent" refers to the underlying factors of variation encoded in the internal low-dimensional representation of the input data.

instance of pre-processing strategy reported to be effective for SHM purposes involves transforming time series into images to highlight *local patterns* that might otherwise be spread over or laying outside the time domain [307,308,309]. This is indeed the strategy we exploit in Chapter 7 within the framework for enhanced Bayesian model updating, to pre-process training datasets of sensor recordings via *MTF encoding*. Moreover, by relying on an AE with a *Siamese architecture trained with contrastive learning*, we enable an informed distance metric within the latent space that reflects the SHM parameters of interest.

307. Giglioni *et al.*, *Deep Autoencoders for Unsupervised Damage Detection with Application to the Z24 Benchmark Bridge*. 2023

308. Garcia *et al.*, *Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms*. 2022

309. Mantawy *et al.*, *Convolutional neural network based structural health monitoring for rocking bridge system by encoding time-series into images*. 2022

Determining an *optimal size* for the low-dimensional latent space remains an open question. The prevailing intuition is that this dimension should be larger than the number of factors responsible for the data variability, that in our case coincides with the number of independent parameters included in $\boldsymbol{\mu}$. When $\mathrm{NN}_{\mathrm{ENC}}$ encodes $\mathbf{U}$ into the corresponding low-dimensional latent representation $\mathbf{h}$, it implicitly addresses an *inverse problem* often lacking a one-to-one correspondence between $\mathbf{U}$ and $\mathbf{h}$. A *redundant* $\mathbf{h}$ allows for multiple equivalent solutions, also helping the training algorithm in balancing the potential poorly initialization of some learnable parameters with the proper initialization of others [310].

310. Lopez-Paz *et al.*, *Easing non-convex optimization with neural networks*. 2018

If an AE employs only *identity functions* as activation functions and the MSE is used as the training loss function, the mappings operated by $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{DEC}}$ can be interpreted as *linear operators*. In this light, the MSE loss function can be viewed from the perspective of the *Schmidt-Eckart-Young theorem* [236], see Eq. (3.13). This insight suggests that in the case of a *linear AE*, using an MSE loss function and imposing an orthonormality constraint on the columns of the weights matrices underlying $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{DEC}}$, the latent space represented by $\mathbf{h}$ tends to the subspace spanned by the POMs. Consequently, the resulting $\mathrm{NN}_{\mathrm{DEC}}$ and $\mathrm{NN}_{\mathrm{ENC}}$ tend to the POD basis matrix and its transpose, respectively[†] [311].

236. Eckart *et al.*, *The approximation of one matrix by another of lower rank*. 1936

[†]It is worth noting that, due to the equivalence of different rotations of the same latent space for reconstruction, the decoder and the encoder do not strictly tend to the orthogonal matrix formed by the first normalized eigenvectors of the sample covariance matrix and its transpose, respectively. Instead, they tend to two matrices obtained by appending the proper orthogonal decomposition basis matrix with an invertible transformation, such as rotation and scaling, and by prefixing its transpose with the inverse of this transformation, respectively.

311. Rolinek *et al.*, *Variational Autoencoders Pursue PCA Directions (by Accident)*. 2019

4.3 PROBABILISTIC GRAPHICAL MODELS

PGMs provide a powerful framework for *reasoning* and *decision-making* under uncertainty — a core problem for enabling DTs of structures in an SHM context. PGMs represent *joint probability distributions*, where *nodes* denote *random variables*, and *edges* connecting nodes represent *direct probabilistic interactions* between them [161]. PGMs offer a series of advantages over the flat expansion of a joint probability distribution [312]:

161. Koller *et al.*, *Probabilistic Graphical Models: Principles and Techniques*. 2009

312. Sucar, *Probabilistic Graphical Models: Principles and Applications*

I PGMs offer a compact representation of probability distributions, even when the explicit representation is large.

II PGMs *factorize* joint distributions over variables by exploiting *local conditional independence structures* and Bayes' rule.

The concept of representing interactions between random variables using a graph structure originated in various communities with different motivations. Early contributions to methods for decision-making under uncertainty, were based on the *naive Bayes model* [313], which later evolved into the *Bayesian network* framework by Judea Pearl [314]. Recent applications of PGMs include contributions to push forward SHM and predictive maintenance strategies [315,316,317,318,319,320,321,322,323], and DT approaches to engineering systems [11,17,19,20,239,324].

The remainder of this Section is organized as follows. First, we briefly introduce the graph formalism in Sec. 4.3.1. Then, in Sec. 4.3.2, we describe the Bayesian network model based on directed graphs. Moving on to Sec. 4.3.3, we extend the static Bayesian network framework to settings where variables are no longer fixed in time. Our aim is to model a system whose state evolves over time, requiring probability distributions over entire trajectories. We thus introduce the *dynamic Bayesian network* (DBN) model, which allows us to represent systems evolving over time. In Sec. 4.3.4, we approach sequential decision making under uncertainty, with reference to Markov decision processes. Finally, in Sec. 4.3.5, we extend the DBN framework to forms of reasoning where we can interact with an *external environment* via *informed actions*. References for the equations employed in the following can be found in [161,162,325].

4.3.1 Graph formalism

A PGM is a model using a graph to express the conditional dependence structure between random variables. A *graph* is a data structure $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ consisting of a set of $N_{\mathcal{X}} \in \mathbb{N}$ nodes $\mathcal{X} = \{X_{N_1}, \dots, X_{N_{\mathcal{X}}}\}$ and a set of edges $\mathcal{E}$. A pair of nodes $X_{N_i}, X_{N_j} \in \mathcal{X}$ can be connected by a *directed edge* $X_{N_i} \rightarrow X_{N_j} \in \mathcal{E}$ or $X_{N_j} \leftarrow X_{N_i} \in \mathcal{E}$, or by an *undirected edge* $X_{N_i} - X_{N_j} \in \mathcal{E}$. A graph is directed if all its edges are directed, and undirected if all its edges are undirected. If $X_{N_i} \rightarrow X_{N_j}$, X_{N_j} is said a *child* of X_{N_i}, and X_{N_i} is said a *parent* of X_{N_j}. We denote the parents of X_{N_i} as $\overline{\mathrm{Pa}}(X_{N_i}) \in \mathcal{X}$, and its children as $\underline{\mathrm{Ch}}(X_{N_i}) \in \mathcal{X}$.

A graph $\mathcal{G}$ may be specified by a set of $N_{\mathcal{Y}} \in \mathbb{N}$ *local functions*, or *local factors*, $\phi_{n_{\mathcal{Y}}}(\mathcal{Y}_{n_{\mathcal{Y}}})$ of some subset $\mathcal{Y}_{n_{\mathcal{Y}}}$ of $\mathcal{X}$, with $n_{\mathcal{Y}} = 1, \dots, N_{\mathcal{Y}}$, representing the associated joint probability

313. Warner *et al.*, *A Mathematical Approach to Medical Diagnosis: Application to Congenital Heart Disease*. 1961

314. Pearl, *Probabilistic Reasoning in Intelligent Systems*. 1988

315. Andriotis *et al.*, *Managing engineering systems with large state and action spaces through deep reinforcement learning*. 2019

316. Morato *et al.*, *Inference and dynamic decision-making for deteriorating systems with probabilistic dependencies through Bayesian networks and deep reinforcement learning*. 2023

317. Papakonstantinou *et al.*, *Optimum inspection and maintenance policies for corroded structures using partially observable Markov decision processes and stochastic, physically based models*. 2014

318. Papakonstantinou *et al.*, *Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part I: Theory*. 2014

319. Papakonstantinou *et al.*, *Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation*. 2014

320. Hlaing *et al.*, *Inspection and maintenance planning for offshore wind structural components: integrating fatigue failure criteria with Bayesian networks and Markov decision processes*. 2022

321. Hughes *et al.*, *A probabilistic risk-based decision framework for structural health monitoring*. 2021

322. Morato *et al.*, *Optimal inspection and maintenance planning for deteriorating structural components through dynamic Bayesian networks and Markov decision processes*. 2022

323. Andriotis *et al.*, *Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints*. 2021

11. Li *et al.*, *Dynamic Bayesian Network for Aircraft Wing Health Monitoring Digital Twin*. 2017

17. Arcieri *et al.*, *Bridging POMDPs and Bayesian decision making for robust maintenance planning under model uncertainty: An application to railway systems*. 2023

distribution, as follows:

$$p(X_{N_1}, \ldots, X_{N_\mathcal{X}}) = \prod_{n_\mathcal{Y}=1}^{N_\mathcal{Y}} \phi_{n_\mathcal{Y}}(\mathcal{Y}_{n_\mathcal{Y}}). \tag{4.28}$$

There are two classes of problem associated with PGMs:

I *Inference.* The aim is to obtain the marginal or conditional probabilities of a subset of variables given the others.

II *Learning.* The aim is to obtain the graph structure or local factors given observational data for a subset of variables.

19. Kapteyn *et al.*, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*. 2021

20. Sisson *et al.*, *Digital Twin Approach for Component Health-Informed Rotorcraft Flight Parameter Optimization*. 2022

239. Torzoni *et al.*, *A digital twin framework for civil engineering structures*. 2024

324. Karve *et al.*, *Digital twin approach for damage-tolerant mission planning under uncertainty*. 2020

161. Koller *et al.*, *Probabilistic Graphical Models: Principles and Techniques*. 2009

162. Russell, *Artificial Intelligence: A Modern Approach*. 2020

325. Sutton *et al.*, *Reinforcement Learning: An Introduction*. 2018

4.3.2 BAYESIAN NETWORKS

Bayesian networks, also known as *belief networks*, *probabilistic networks*, *causal networks*, or *knowledge maps*, are a specific type of PGM. They are characterized by a *directed acyclic* structure* in which nodes represent random variables and edges connecting nodes represent conditional dependencies between variables. *Causal relationships* between variables are described using *conditional probability tables* (CPTs) for discrete random variables†, and conditional PDFs for continuous random variables.

*A directed acyclic graph is a graph characterized by the absence of directed cycles.

†In a conditional probability table, each column contains the conditional probability of each node value given the parent nodes. Each column sums to one since the entries represent an exhaustive set of cases for the variable. A node with no parents has only one row, indicating the prior probabilities of each possible value. Conditional probability tables represent the a-priori knowledge regarding the random variables. They can be viewed as a set of parameters, to be determined based on historical data, accumulated experience, or learned online from data.

In a Bayesian network $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$, each node $X_{N_i} \in \mathcal{X}$, with $N_i = N_1, \ldots, N_\mathcal{X}$, has a conditional probability distribution $p(X_{N_i}|\overline{\text{Pa}}(X_{N_i}))$ quantifying the influence of the parents on the node. By denoting $\text{NDe}(X_{N_i}) \in \mathcal{X}$ the variables in the graph that are not *descendants* of X_{N_i}, the Bayesian network encodes a set of conditional *local independence* assumptions $\{X_{N_i} \perp \text{NDe}(X_{N_i})|\overline{\text{Pa}}(X_{N_i})\}_{N_i=N_1}^{N_\mathcal{X}}$. These assumptions state that X_{N_i} is conditionally independent of its non-descendants given its parents. The local factors in Eq. (4.28) correspond precisely to the conditional probabilities $p(X_{N_i}|\overline{\text{Pa}}(X_{N_i}))$ implied by the joint distribution. Thus, Eq. (4.28) can be reformulated as:

$$p(X_{N_1}, \ldots, X_{N_\mathcal{X}}) = \prod_{N_i=N_1}^{N_\mathcal{X}} p(X_{N_i}|\overline{\text{Pa}}(X_{N_i})). \tag{4.29}$$

Accordingly, a Bayesian network is the skeleton for compactly expressing a joint distribution in a factorized way, through the associated conditional independence assumptions [162].

A simple example of Bayesian network is shown in Fig. 4.5a. Here, bold outlines denote *observed quantities*, while thin outlines denote *quantities to be inferred*. Since X_{N_1} is independent of other nodes, its state is specified by the unconditional distribution $X_{N_1} \sim p(x_{N_1})$. X_{N_1} is a parent of X_{N_2}, and the arrow $X_{N_1} \rightarrow X_{N_2}$ denotes that X_{N_1} has a direct influence on X_{N_2}. Even if X_{N_2} is

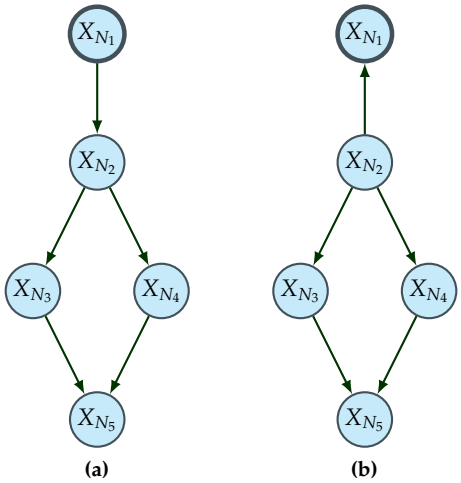


Figure 4.5: Exemplary Bayesian networks for: (a) causal reasoning and (b) evidential reasoning. Nodes with bold outlines denote observed random variables, while nodes with thin outlines denote unobserved quantities to be estimated. Directed solid edges represent variables dependencies encoded via conditional probability distributions.

a parent of both X_{N_3} and X_{N_4}, X_{N_3} and X_{N_4} are conditionally independent given X_{N_2}. Since X_{N_1} is the only observed variable, any change in X_{N_1} has an effect on X_{N_2}, that affects X_{N_3} and X_{N_4}, and so on. This is a *causal trail*, in which the posterior distribution over the Bayesian network can be derived via uncertainty propagation:

$$p(X_{N_2}, X_{N_3}, X_{N_4}, X_{N_5} | X_{N_1} = x_{N_1}) = \phi_2 \phi_3 \phi_4 \phi_5, \tag{4.30}$$

conveniently factorized according to Eq. (4.29), with factors:

$$\phi_2 = p(X_{N_2} | X_{N_1} = x_{N_1}), \tag{4.31}$$
$$\phi_3 = p(X_{N_3} | X_{N_2}), \tag{4.32}$$
$$\phi_4 = p(X_{N_4} | X_{N_2}), \tag{4.33}$$
$$\phi_5 = p(X_{N_5} | X_{N_3}, X_{N_4}). \tag{4.34}$$

If X_{N_2} was also observed, any change in X_{N_1} would not affect X_{N_2}, as we already know its value. In this case, any change in X_{N_3} does not affect X_{N_4}, since this latter only depends on X_{N_2}. In a different scenario, if X_{N_3} was observed instead of X_{N_2}, influence would flows from X_{N_3} to X_{N_4} through the *common cause structure* $X_{N_3} \leftarrow X_{N_2} \rightarrow X_{N_4}$, active if and only if X_{N_2} is not observed. On the other hand, when X_{N_5} is not observed, any change in X_{N_3} reflects some change in X_{N_5} but not in X_{N_4}. However, when X_{N_5} is observed, the *common effect structure* $X_{N_3} \rightarrow X_{N_5} \leftarrow X_{N_4}$ is active, allowing influence to flow from X_{N_3} through X_{N_5} to affect the belief about X_{N_4}

A different situation is the one associated with the Bayesian network in Fig. 4.5b. In this case, there exists an *evidential trail* due to the reversed edge $X_{N_2} \rightarrow X_{N_1}$, given that X_{N_1} is observed.

When data of any child node are obtained, a Bayesian network is updated through Bayesian inference. Specifically, we have:

$$
\begin{aligned}
&p(X_{N_2}, X_{N_3}, X_{N_4}, X_{N_5} | X_{N_1} = x_{N_1}) \propto \\
&\quad \propto p(X_{N_1} = x_{N_1} | X_{N_2}, X_{N_3}, X_{N_4}, X_{N_5}) p(X_{N_2}, X_{N_3}, X_{N_4}, X_{N_5}) = \\
&\quad = \phi_2 \phi_3 \phi_4 \phi_5,
\end{aligned}
\tag{4.35}
$$

with factors:

$$\phi_2 = p(X_{N_1} = x_{N_1} | X_{N_2}) p(X_{N_2}), \tag{4.36}$$

$$\phi_3 = p(X_{N_3} | X_{N_2}), \tag{4.37}$$

$$\phi_4 = p(X_{N_4} | X_{N_2}), \tag{4.38}$$

$$\phi_5 = p(X_{N_5} | X_{N_3}, X_{N_4}), \tag{4.39}$$

where $p(X_{N_1} = x_{N_1} | X_{N_2})$ is a likelihood function, and the product $p(X_{N_2})\phi_3\phi_4\phi_5$ is the prior distribution over X_{N_2}, X_{N_3}, X_{N_4}, and X_{N_5}. Despite the computational complexity increases exponentially with the size of the graph, efficient inference algorithms can be applied to obtain posterior distributions over the unobserved variables, also called *hidden state variables*. If the spaces of the unobserved variables are *discrete*, we can propagate and update the relative belief exactly with a single pass of the *sum-product message-passing algorithm* [326]. When exact inference methods can not be employed, sampling-based *approximate inference* methods can be alternatively exploited.

326. Pearl, *Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach*. 1982

The Bayesian network model forms the foundation for the PGM used in Chapter 8 for the digital twinning of engineering systems. To achieve the dynamic decision network model we actually employ, in the following sections, we extend the Bayesian network model to *time-dependent* contexts. Moreover, we equip it with *decisions* and *utility* nodes, enabling the interaction with an external environment through informed actions.

4.3.3 DYNAMIC BAYESIAN NETWORKS

The Bayesian network model introduced in Sec. 4.3.2 is well-suited for *static* contexts. For *time-dependent scenarios*, a DBN allows us to construct a single compact model that captures the dynamics of a system and yields distributions over different *trajectories*. A DBN can be seen as a series of static Bayesian networks, one for each time slice. While static Bayesian networks only feature *intra-time-slice* edges, connecting variables in the same time slice, a DBN also features additional *inter-time-slice* edges connecting variables in different time slices.

The goal of a DBN $\mathcal{G} = \langle \mathcal{X}, \mathcal{E} \rangle$ is to represent a joint distribution, assigning a value to each variable for each possible time. We

first consider a *time discretization*, and denote non-dimensional time steps by t. The physical time duration between successive time slices depends on the problem, and it is assumed to be regularly spaced with granularity equal to the time between two consecutive observations.

By denoting the set of variables in $\mathcal{G}$ at time t as $\mathcal{X}^{(t)} = \{X_{N_1}^{(t)}, \dots, X_{N_\mathcal{X}}^{(t)}\}$, the associated distribution over a time partition $t = 0, \dots, T$, with finite horizon $T > 0$, is denoted as:

$$p(\mathcal{X}^{(0:T)}) = p(X_{N_1}^{(0:T)}, \dots, X_{N_\mathcal{X}}^{(0:T)}), \tag{4.40}$$

where $X_{N_i}^{(t_1:t_2)}$ is the sequence of variables $\{X_{N_i}^{(t)}\}_{t=t_1}^{t_2}$, with $N_i = N_1, \dots, N_\mathcal{X}$. By means of the chain rule for probabilities, we can rewrite the distribution in Eq. (4.40) as:

$$p(\mathcal{X}^{(0:T)}) = p(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} p(\mathcal{X}^{(t+1)} | \mathcal{X}^{(0:t)}). \tag{4.41}$$

By assuming that the DBN represents a *first-order Markov process*, each variable in $\mathcal{X}^{(t)}$ can have parents only in the (t)-th time slice or in the immediately preceding one. Thanks to the corresponding conditional independence structure, we can define a more compact representation of $p(\mathcal{X}^{(0:T)})$, as follows:

$$p(\mathcal{X}^{(0:T)}) = p(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} p(\mathcal{X}^{(t+1)} | \mathcal{X}^{(t)}), \tag{4.42}$$

allowing us to represent the probability distribution over different trajectories, given the *initial state distribution* $p(\mathcal{X}^{(0)})$ and the *transition model* $p(\mathcal{X}^{(t+1)} | \mathcal{X}^{(t)})$. For our digital twinning purposes, assuming that the DBN represents a Markov process implies that the modeled state transition probabilities only consider asset-twin interactions within the current time slice or the immediately preceding one. Moreover, it involves that the actions to be taken on the asset are informed solely by the observational data for the current time slice, rather than the full observation history. Despite these limitations, the Markov assumption is crucial for making the digital twinning PGM we employ in Chapter 8 computationally tractable.

The last assumption we make is that the underlying Markov process is *stationary*, meaning that the transition $p(\mathcal{X}^{(t+1)} | \mathcal{X}^{(t)})$ is the same for any t. For our digital twinning purposes, this implies that the resulting PGM is mostly suited for capturing a physical state evolving according to a stationary process. While this aspect does not limit the capabilities of our DT to correctly track *non-stationary* evolutions of the asset, in Chapter 8, we

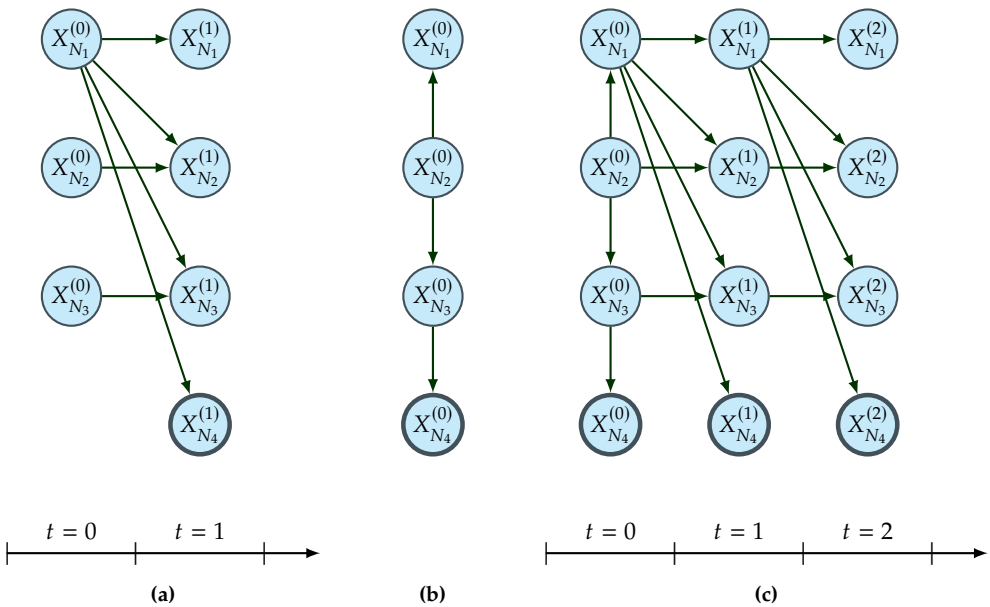


Figure 4.6: Exemplary dynamic Bayesian networks: (a) two-time-slice Bayesian network for the process, (b) Bayesian network representing the initial distribution over random variables, (c) resulting dynamic Bayesian network unrolled over three time slices. Nodes with bold outlines denote observed random variables, while nodes with thin outlines denote unobserved quantities to be estimated. Directed solid edges represent variables dependencies encoded via conditional probability distributions.

envision a possible research line to enable more personalized and robust DTs by removing this assumption.

The set of assumed conditional dependencies between random variables allows us to specify the graph topology from two time slices, and then unroll it for any desired number of time slices. The factorization in Eq. (4.42) is then achieved by representing the transition model $p(\mathcal{X}^{(t+1)}|\mathcal{X}^{(t)})$ using a Bayesian network. As an example, Fig. 4.6a shows a *two-time-slice Bayesian network* for a process over $\mathcal{X}$. It is a conditional Bayesian network over $\mathcal{X}^{(1)}$, given a set of *interface* variables $\mathcal{X}_I \subseteq \mathcal{X}$. In particular, $\mathcal{X}_I$ includes the variables in $\mathcal{X}^{(0)}$ with a direct effect on $\mathcal{X}^{(1)}$.

Given a Bayesian network $\mathcal{G}^0$ representing the initial distribution over random variables $\mathcal{X}^{(0)}$, as shown in Fig. 4.6b, a DBN is defined as a pair $\langle \mathcal{G}^0, \mathcal{G}^{\rightarrow} \rangle$, where $\mathcal{G}^{\rightarrow}$ is a two-time-slice Bayesian network for the process. For any desired time horizon $T > 0$, the distribution over $\mathcal{X}^{(0:T)}$ is defined as the unrolled Bayesian network (see Fig. 4.6c) where, for any $N_i = N_1, \ldots, N_{\mathcal{X}}$:

- The structure and conditional probabilities governing the causal relationships of $X^{(0)}_{N_i}$ are as those for X_{N_i} in $\mathcal{G}^0$.
- The structure and conditional probabilities governing the causal relationships of $X^{(t>0)}_{N_i}$ are as those for $X^{(1)}_{N_i}$ in $\mathcal{G}^{\rightarrow}$.

Given a sequence of observations, a DBN can be unrolled until it fully encompasses the obtained data. Additional time slices beyond the last observation have no effect on inference within the observation period. The unrolled DBN becomes a

standard Bayesian network, and beliefs can be updated and propagated as for static Bayesian networks.

4.3.4 UTILITY AND SEQUENTIAL DECISION MAKING

In this Section, we move to the task of interacting with an external *environment* through actions. In a *decision-making* setting, an *agent* must choose from a set of possible *actions*, each potentially leading to uncertain effects on the *state* of the *system*. The decision-making process aims at maximizing, at least on average, the numerical *utilities* assigned to each possible action outcome. This involves taking into account both the probabilities associated with various outcomes and our preferences among them.

When faced with a decision, the agent must choose an action u from a set $\mathcal{U}$ of $N_{\mathcal{U}} \in \mathbb{N}$ possible actions. This decision-making process relies on the knowledge about state variables. Let $\mathcal{S}$ represent the set of $N_{\mathcal{S}} \in \mathbb{N}$ states that a given system could assume. The system state S is a random variable distributed as $S \sim p(s)$*, where $p(s)$ encodes the relative likelihood that $S = s$.

*The state of a system, component, or process, is generally not directly observable. It is usually only partially and indirectly observable through observational data.

A *utility function* $\mathbb{U}(s, u) : \mathcal{S} \times \mathcal{U} \mapsto \mathbb{R}$ encodes the preferences of an agent for the joint result of taking an action u while being in state s. In an uncertain context, the perceived benefit of an action u is measured by the *expected utility* $\overline{\mathbb{U}}(u) = \mathbb{E}[\mathbb{U}(S, u)]$, where $\overline{\mathbb{U}}(u) : \mathcal{U} \mapsto \mathbb{R}$ represents the average utility an agent expects from taking action u, and is given as:

$$\overline{\mathbb{U}}(u) = \mathbb{E}[\mathbb{U}(S, u)] = \sum_{s \in \mathcal{S}} \mathbb{U}(s, u) p(s). \tag{4.43}$$

The *principle of maximum expected utility* asserts that a *rational agent* should choose the optimal, or rational, action $u^* \in \mathcal{U}$ that maximizes the expected utility, as follows:

$$u^* = \arg\max_{u \in \mathcal{U}} \mathbb{E}[\mathbb{U}(S, u)]. \tag{4.44}$$

Sequential decision problems. In *sequential decision problems*, the agent's utility is influenced by a *sequence of decisions*. We consider a time discretization using non-dimensional time steps, and denote the system state at time t as $s_t \in \mathcal{S}$, with $S_t \sim p(s_t)$. Moreover, we denote the control input at time t as $u_t \in \mathcal{U}$. We define a *control policy* $\pi : \mathcal{S} \mapsto \mathcal{U}$ as the mapping from any system state to the space of actions. In sequential decision problems, the goal is to find the *optimal control policy* $\pi^*(S_t)$ that provides the optimal action u_t^* for each possible state s_t.

The optimal policy $\pi^*(S_t)$ is learned by looking for the action u_t^* that maximizes the expected utility over a *planning horizon*

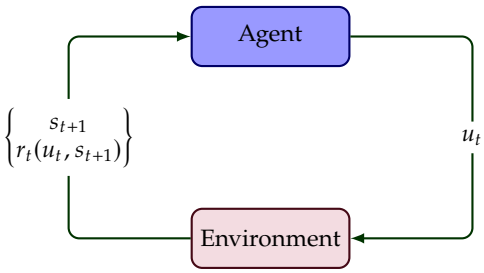


Figure 4.7: Agent-environment interaction: schematic representation.

$(0, T)$. To measure the expected utility, we refer to a set of *rewards* $\mathcal{R}$, quantifying the value associated with each possible set $\{s_t, u_t, s_{t+1}\}$. In this Thesis, we assume that the reward function is independent on the current system state, namely $r_t = r_t(u_t, s_{t+1})$. This reward function can encode various forms of success or failure for the system, as well as the costs associated with applying the control inputs. The last component required to define a sequential decision problem is a *Markov transition model* $p(s_{t+1}|s_t, u_t) : \mathcal{S} \times \mathcal{S} \times \mathcal{U} \mapsto [0, 1]$, similar to those employed in Sec. 4.3.3. This transition model is a CPT that encodes the probability of reaching any state s_{t+1} at time $t + 1$, given the current state s_t and an action taken u_t. For every possible action, there is a corresponding transition probability matrix encoding the conditional probabilities for all the combinations of states.

The sequential decision problem can be viewed from the perspective of an *agent-environment interaction*, as depicted in Fig. 4.7. In this scenario, the agent perceives the environment and aims to maximize the *long-term accumulation of rewards* by choosing an action u_t that influences the environment at time $t + 1$. The environment interacts with the agent by defining the evolution of the system state, and providing a reward $r_t(u_t, s_{t+1})$ for taking an action u_t and moving to the next state s_{t+1}.

Markov decision processes. *Markov decision processes* (MDPs) provide a framework for describing sequential decision problems in *fully observable*†, stochastic environments with Markov transition models and *additive* rewards [327,328]. Formally, an MDP is a 4-tuple $\langle \mathcal{S}, \mathcal{U}, p(s_{t+1}|s_t, u_t), \mathcal{R} \rangle$, comprising the space of system states, the set of actions, the transition model‡, and the space of rewards. The problem of finding the optimal control policy is inherently stochastic. Consequently, the associated objective function is additive and relies on expectations [318]. This is typically expressed as the *total expected discounted reward* over the planning horizon $(0, T)$.

One way to characterize an MDP is to consider the expected utility associated with a policy π when starting in any state s_0 and following π thereafter. To this aim, the *state-value function*

†Standard Markov decision processes assume full observability of the environment, meaning that the agent has a complete and exact knowledge of the current state at all times.

327. Bellman, *A Markovian Decision Process*. 1957

328. Howard, *Dynamic programming and Markov processes*. 1960

‡The availability of a transition model for a Markov decision process influences the selection of appropriate solution algorithms. Dynamic programming algorithms, for instance, require an explicit model. In situations where representing the transition model becomes challenging, a simulator is often employed to implicitly model the Markov decision process. This simulator allows for sampling environment trajectories from the underlying transition model. This is the case of episodic reinforcement learning tasks, where an environment simulator is queried with control actions to advance the system state.

318. Papakonstantinou *et al.*, *Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part I: Theory*. 2014

$\mathbb{V}^{\pi}(S_t) : \mathcal{S} \mapsto \mathbb{R}$ quantifies, for every state s_t, the amount of discounted reward the decision-maker can gather by starting in s_0 and acting according to π, as follows:

$$\mathbb{V}^{\pi}(s_t) = \sum_{t=0}^{T} \gamma^t \mathbb{E}_{\pi}[r_t(u_t, S_{t+1})|s_0], \tag{4.45}$$

where $\gamma \in (0, 1]$ is a *discount factor* to obtain the present value, and $T = +\infty$ for an *infinite planning horizon*. In Eq. (4.45), $\mathbb{E}_{\pi}$ denotes the expected value of a random variable given that the agent follows policy π, and the probability of each state s_t at time $t > 0$ is estimated using the transition model $p(s_{t+1}|s_t, u_t)$.

For any control policy π and for each possible state s_t, the state-value function satisfies the following recursive relationship between the value of s_t and the value of its possible successor states s_{t+1}:

$$\mathbb{V}^{\pi}(s_t) = \sum_{u_t \in \mathcal{U}} \pi(u_t|s_t) \sum_{s_{t+1} \in \mathcal{S}} p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}^{\pi}(s_{t+1})\big). \tag{4.46}$$

Eq. (4.46) is the *Bellman equation* for $\mathbb{V}^{\pi}$, which states that the value of any state s_t is equal to the discounted value for the expected next state s_{t+1}, plus the expected reward from s_{t+1}.

Although there could be more than one optimal policy $\pi^*(S_t)$, they all share the same *optimal state-value function* $\mathbb{V}^{\pi*}(s_t)$. For any given state s_t, an optimal control policy $\pi^*(S_t)$ satisfies:

$$\mathbb{V}^{\pi*}(s_t) = \max_{\pi} \mathbb{V}^{\pi}(s_t). \tag{4.47}$$

According to the principle of maximum expected utility, for any given state s_t, acting following an optimal policy $\pi^*(S_t)$ maximizes the expected utility of the subsequent state, thus:

$$\pi^*(s_t) = \arg\max_{u_t \in \mathcal{U}} \left(\sum_{s_{t+1} \in \mathcal{S}} p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}^{\pi*}(s_{t+1})\big) \right), \tag{4.48}$$

where $\mathbb{V}^{\pi*}(s_{t+1})$ measures the utility associated with s_{t+1} when following π^*. Eq. (4.48) highlights the challenge in finding $\pi^*(S_t)$, as it appears on both sides of the equality.

For an infinite time horizon, the policy is *stationary* and depends only on the state s_t. In contrast, for a finite time horizon, the policy is *non-stationary* since it depends on both the current state s_t and time t. For our digital twinning purposes, we focus on infinite horizon policies. However, we recognize that this assumption is limiting and unrealistic for civil structures. Therefore, in Chapter 8, we propose a possible research line to

extend our DT framework by incorporating a finite planning horizon that represent the design lifetime of the asset.

Referring to $\pi^*(S_t)$, the Bellman equation can be written as:

$$\mathbb{V}^{\pi*}(s_t) = \max_{u_t \in \mathcal{U}} \left(\sum_{s_{t+1} \in \mathcal{S}} p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}^{\pi*}(s_{t+1})\big) \right), \tag{4.49}$$

which is called *Bellman optimality equation*. This equation is actually a system of $N_{\mathcal{S}}$ equations in $N_{\mathcal{S}}$ unknowns, one for each state of the system. The optimal state-value functions are the unique solutions to these equations. Once $\mathbb{V}^{\pi*}(s_t)$ is known, an optimal policy can be immediately determined and the actions that are optimal after one-step-ahead search coincide with the long-term optimal actions. However, the optimal action $u_t^*(s_t)$ is selected through the max operator, which makes the *system of Bellman equations* nonlinear.

Value iteration algorithm. *Dynamic programming* is an algorithmic paradigm developed by Bellman [329]. It involves simplifying a problem by breaking it down into simpler sub-problems in a recursive manner. Dynamic programming provides a family of solution techniques for computing optimal policies given a perfect model of the MDP environment. Among them, one of the most common is the *value iteration algorithm*, also known as *backward induction*. This algorithm is employed within our DT framework presented in Chapter 8 to solve the planning problem induced by the PGM and find a health-dependent control policy.

329. Bellman, *Dynamic Programming*. 1957

The underlying idea of value iteration is to calculate the utility of each state and then use them to select an optimal action. Given that $\mathbb{V}^{\pi*}(s_t)$ appears on both sides of Eq. (4.49), the algorithm proceeds recursively using the *Bellman update step*:

$$\mathbb{V}^{\pi}(s_t) \leftarrow \max_{u_t \in \mathcal{U}} \left(\sum_{s_{t+1} \in \mathcal{S}} p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}^{\pi}(s_{t+1})\big) \right), \tag{4.50}$$

which turns the Bellman optimality equation into an update rule. In detail, we start by initializing $\mathbb{V}^{\pi}(s_t) = 0$, then we calculate the right-hand side of Eq. (4.50), and plug it into the left-hand side to simultaneously update the utility of each state from the utilities of its neighbors. Formally, after an infinite number of iterations, the algorithm converges to a steady-state equilibrium. In practice, we stop the procedure once the change in the state-value function of each state in a sweep is smaller than a prescribed tolerance $\epsilon_{\mathbb{V}} \in \mathbb{R}$. The value iteration algorithm is formalized in Algorithm 4.1.

Algorithm 4.1 Value iteration

Input: states space $\mathcal{S}$
actions space $\mathcal{U}$
rewards space $\mathcal{R}$
transition model $p(s_{t+1}|s_t, u_t)$
discount factor γ
tolerance on the value error for any state $\epsilon_{\mathbb{V}}$

Initialize the expected utilities $\mathbb{V}'(s_t) = 0, \forall s_t \in \mathcal{S}$
Initialize $\Delta_{\mathbb{V}} > \epsilon_{\mathbb{V}}$
while $\Delta_{\mathbb{V}} \geq \epsilon_{\mathbb{V}}$ **do**
    $\Delta_{\mathbb{V}} = 0$
    for $s_t \in \mathcal{S}$ **do**
        $\mathbb{V}(s_t) \leftarrow \mathbb{V}'(s_t)$
        $\mathbb{V}'(s_t, u_t) =$
            $\max_{u_t \in \mathcal{U}} \sum_{s_{t+1} \in \mathcal{S}} p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}(s_{t+1})\big)$
        $\Delta_{\mathbb{V}} = \max(\Delta_{\mathbb{V}}, |\mathbb{V}(s_t) - \mathbb{V}'(s_t)|)$
return deterministic control policy $\pi \approx \pi^*$:
        $\pi(S_t) = \arg\max_{u_t \in \mathbb{U}} \Big(p(s_{t+1}|s_t, u_t)\big(r_t(u_t, s_{t+1}) + \gamma \mathbb{V}(s_{t+1})\big)\Big)$

4.3.5 DYNAMIC DECISION NETWORKS

In this Section, we finally move to *decision networks*, or *influence diagrams*, which synthesize the problem of determining an optimal policy based on beliefs about state variables by integrating Bayesian networks with decisions and utilities. Decision networks are directed acyclic graphs, where random variables are depicted as *circle nodes*, decisions as *square nodes*, and utilities as *diamond nodes* (that must not have children). Edges connecting random variables or decision variables to utility nodes encode a dependency of the utility function on these variables.

An exemplary decision network is depicted in Fig. 4.8a. In this graph, *Bold outlines* denote observed quantities, while *thin outlines* denote estimated quantities. *Solid edges* represent the variables' dependencies encoded via conditional probability distributions, while *dashed edges* represent variables' dependencies encoded via deterministic functions. The system state $S \in \mathcal{S}$ is independent of other nodes, thus its value is determined by the unconditional distribution $S \sim p(s)$. Both $U \in \mathcal{U}$ and $U^A \in \mathcal{U}$ represent decision variables. $U \sim p(u)$ is depicted with a circle node, representing the *predicted* belief about the optimal action, while $U^A = u^A$ is depicted with a square node, representing the *actual* control effectively taken. The reward variable $R \sim p(r)$ quantifies the expected utility associated with the PGM.

The solid edge $S \rightarrow U$ denotes a dependency encoded through a CPT. This CPT corresponds to an S-dependent control policy, mapping the belief over S onto control actions U. In

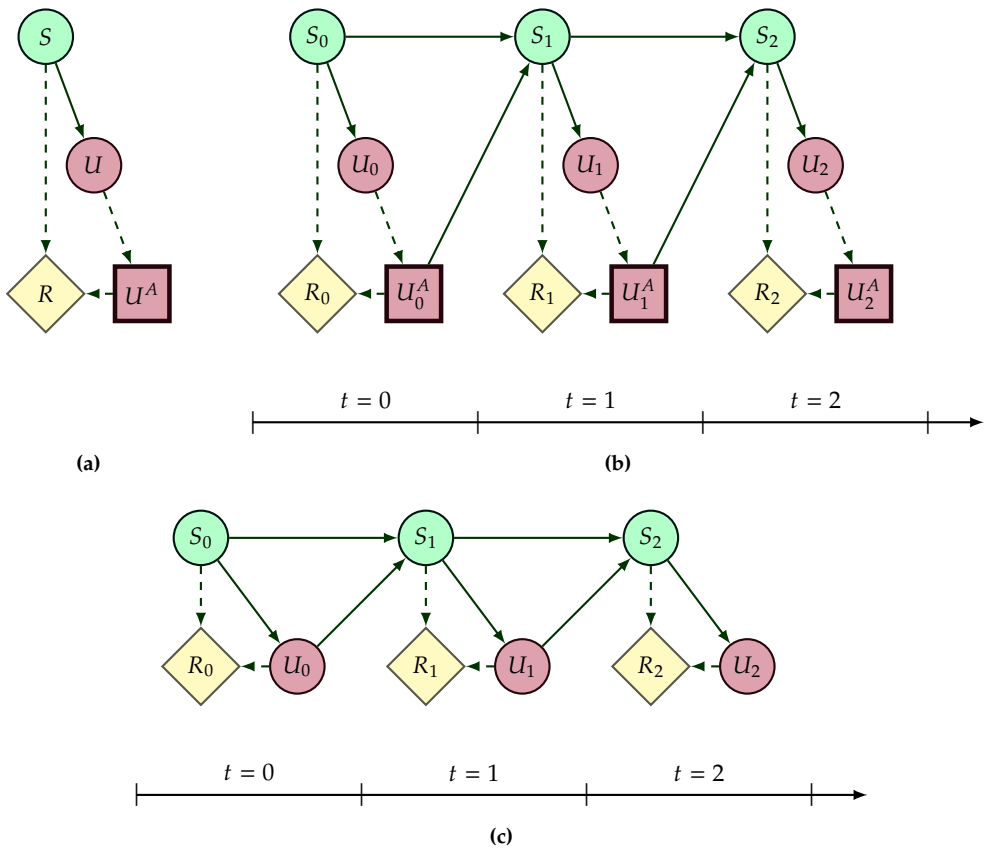


Figure 4.8: Exemplary decision networks: (a) static decision network; (b) dynamic decision network unrolled over three time slices; (c) corresponding dynamic Bayesian network. Circle nodes denote random variables, square nodes denote actions, and diamond nodes denote the objective function. Nodes with bold outlines denote observed random variables, while nodes with thin outlines denote unobserved quantities to be estimated. Directed solid edges represent variables dependencies encoded via conditional probability distributions, while directed dashed edges represent variables' dependencies encoded via deterministic functions.

contrast, the edge $U \rightarrow U^A$ is depicted as dashed, due to the following reason. In a decision network, the agent can actively select the value of a decision node, thereby influencing the values of downstream variables. Since the agent knows exactly which action it takes on the system, once the decision node is set, it behaves like an evidence variable. In the case of a direct dependence $S \rightarrow U^A$, this evidence would update backward the state S, changing its posterior estimate based on the chosen control input $U^A = u^A$. However, this is not the desired behavior in view of applying decision networks for our digital twinning purposes. To address this drawback, the control component is split into the *predicted* and *actual* control variables. The dashed connection $U \rightarrow U^A$ denotes a conversion from probabilistic to deterministic control by selecting the best point estimate of U. Similarly, the two connections $S \rightarrow R \leftarrow U^A$ represent a deterministic reward function quantifying costs relative to system state and control inputs, respectively.

We express the belief state over the graph in Fig. 4.8a as:

$$p(S, U, R | U^A = u^A) \propto p(S)\phi^{\text{control}}\phi^{\text{reward}}, \tag{4.51}$$

with factors:

$$\phi^{\text{control}} = p(U|S), \tag{4.52}$$

$$\phi^{\text{reward}} = p(R|S, U^A = u^A), \tag{4.53}$$

where, ϕ_t^{control} is the S-dependent control policy, and ϕ_t^{reward} encapsulates the computation of the reward function.

Decision networks can be extended to *dynamic* decision networks (DDNs) to handle decision problems evolving over time. The augmentation from a decision network to a DDN follows the same assumptions made to move from Bayesian networks to DBNs: (i) a time discretization; (ii) an underlying first-order Markov process; (iii) a stationary transition model. An example of DDN unrolled over three time slices is depicted in Fig. 4.8b. The associated belief state can be factorized up to a *current* time step t_c, as follows:

$$\begin{aligned} &p(S_{0:t_c}, U_{0:t_c}, R_{0:t_c}|U^A_{0:t_c} = u^A_{0:t_c}) \propto \\ &\qquad p(S_0)\phi_0^{\text{control}}\phi_0^{\text{reward}} \prod_{t=1}^{t_c} \left[\phi_t^{\text{control}}\phi_t^{\text{reward}}\phi_t^{\text{history}}\right], \end{aligned} \tag{4.54}$$

with factors:

$$\phi_t^{\text{control}} = p(U_t|S_t), \tag{4.55}$$

$$\phi_t^{\text{reward}} = p(R_t|S_t, U_t^A = u_t^A), \tag{4.56}$$

$$\phi_t^{\text{history}} = p(S_t|S_{t-1}, U_{t-1}^A = u_{t-1}^A). \tag{4.57}$$

Here, ϕ_t^{history} plays the role of a *predictor forward in time* parameterized by a control-dependent CPT, that describes how the system state is expected to evolve.

It is interesting to note that by removing the U_t^A nodes from the DDN in Fig. 4.8b, we obtain the DBN in Fig. 4.8c. This DBN encodes the MDP associated with the computation of the optimal control policy ruling ϕ_t^{control}. The computation of $\pi^*(S_t)$ involves solving the planning problem induced by the expected evolution of S_t, maximizing the expected reward over the planning horizon. For an infinite planning horizon, this optimization problem can be stated as:

$$\pi^*(S_t) = \arg\max_{\pi} \sum_{t=0}^{+\infty} \gamma^t \mathbb{E}[R_t], \tag{4.58}$$

which can be solved through the value iteration algorithm described in Algorithm 4.1. To this aim, the utility function R_t can be decomposed in a weighted sum of two contributions, as:

$$R_t(U_t, S_t) = R_t^{\text{control}}(U_t) + \chi_R R_t^{\text{state}}(S_t), \tag{4.59}$$

where $R_t^{\text{control}}(U_t)$ and $R_t^{\text{state}}(S_t)$ quantify the rewards relative to control inputs and system state, respectively, and $\chi_R \in \mathbb{R}$ is a weighting factor to tune the behavior of the decision-maker.

Once the DDN in Fig. 4.8b is updated at the current time t_c, the resulting belief state can be be used to feed the DBN in Fig. 4.8c and forecast future values of S_t and U_t via belief propagation. This process can be visualized imagining to unroll the DDN in the out-of-plane direction until a prediction time t_p, from the DBN at time t_c. We remark that, unlike the DDN, the DBN is unrolled in future without effectively taking actions on the system. The control inputs for future time steps are simply predictions made to forecast the evolution of the system. In this scenario, the factorization in Eq. (4.54) can be extended over the prediction horizon associated with t_p, as follows:

$$p(S_{t_c+1:t_p}, U_{t_c+1:t_p}, R_{t_c+1:t_p} | U^A_{0:t_c} = u^A_{0:t_c}) \propto \prod_{t=t_c+1}^{t_p} \left[\phi_t^{\text{control}} \phi_t^{\text{future-}R} \phi_t^{\text{future-}S} \right], \tag{4.60}$$

with factors:

$$\phi_t^{\text{control}} = p(U_t | S_t), \tag{4.61}$$

$$\phi_t^{\text{future-}R} = p(R_t | S_t, U_t), \tag{4.62}$$

$$\phi_t^{\text{future-}S} = p(S_t | S_{t-1}, U_{t-1}), \tag{4.63}$$

where $\phi_t^{\text{future-}S}$ and $\phi_t^{\text{future-}R}$ resemble ϕ_t^{history} and ϕ_t^{reward}, respectively, but they enable future state predictions and probabilistic estimates of the reward, conditioned on *suggested* control inputs that have not yet been selected and applied.

4.4 THESIS PERSPECTIVE

In this Chapter, we have discussed several ML methods employed throughout this Thesis to provide the algorithmic platform that supports physics-based modeling for realizing effective DTs of engineering structures. Their primary goal is to learn relationships between data and quantities of interest. We have started the discussion with the key aspects of ML, highlighting the differences in the rationale behind regression and classification algorithms. Then, we have extensively focused on the DL paradigm, examining the technical aspects related to optimization and regularization, as well as the DL models employed in this Thesis for SHM purposes. Finally, we have analyzed different types of Bayesian networks aimed at the probabilistic modeling of asset-twin interactions.

We envision the contribution of ML algorithms to the objectives reached in this Thesis along three major themes:

I *Surrogate modeling.* The use of ML algorithms, and in particular DL models, is extensively employed in this Thesis to surrogate the input-output map, from input parameters to sensor data or relevant data features. Specifically, data-driven emulators are employed in Chapter 6, within the proposed MF-DNN surrogate modeling strategy, to replace computationally expensive physics-based models. Here, the MF perspective is leveraged for the sequential training of a composition of DNNs designed to map damage and operational parameters onto sensor recordings. The strategy blends small HF datasets with large LF datasets, to reduce the computational burden of supervised training while ensuring the accuracy of the approximation. The MF-DNN-SM consists of a fully-connected LF-DNN, mimicking sensor recordings in the undamaged condition, and an LSTM HF-DNN, adaptively refining the approximation with the effect of damage. Differently, in the framework for enhanced Bayesian model updating proposed in Chapter 7, SMs are exploited to emulate the functional link between parameters to be updated and a low-dimensional feature space by relying on a fully-connected DL model. In both cases, the DL paradigm is crucial for enabling strong modeling capabilities and computational efficiency during evaluation.

II *Data assimilation and inverse problems.* We regard the identification of SHM parameters from sensor measurements as a crucial aspect for enabling DTs of engineering structures. Since observational data depend on the asset state, this involves solving an inverse problem. When approached from a data-driven perspective, this entails learning a mapping from measurements to the sought parameters, that reverses the cause-effect relationship inherent in the underlying structural response or in the forward operator simulated via physics-based models. When employing DL models to approximate this inverse mapping, we integrate the parametrization of damage and operational parameters with the reduced basis method discussed in Chapter 3. This provides a knowledge platform that we exploit offline to efficiently replicate the impact of damage on the structural response while populating training datasets. We systematically employ this procedure for the feature extractors considered in Chapter 5, Sec. 6.3, and Chapter 7, all relying on a *Siamese architecture*. Similarly, the DL classifiers for

damage detection and localization, as well as the DL regressors for damage quantification employed within the DT framework proposed in Chapter 8, also rely on simulation-based supervision and exploit CNNs to automatically select and extract damage-sensitive features from noisy vibration recordings in real time. Despite not using DL models, the same underlying idea of mapping (observable) data features to quantities of interest also characterizes the k-NN regressors employed in Chapter 5 and Sec. 6.3 for real-time damage localization and identification of operational conditions, respectively. Differently, when addressing the SHM inverse problem from a model-based perspective, there is no need for datasets dedicated to the offline learning of a possible inverse mapping. This is exemplified by the MCMC-based approach exploited in Sec. 6.3, where prior knowledge is instead encoded within the MF-DNN-SM used to assess the likelihood function of the sampler. Finally, the hybrid perspective considered in Chapter 7 capitalizes on the strengths of both data-driven and model-based approaches by combining the automatic feature extraction capabilities enabled by a peculiar AE architecture with a probabilistic model-based MCMC framework.

III *Reasoning and decision-making under uncertainty.* The DBN and DDN models presented in this Chapter are the algorithmic platforms we choose to enable the automatic, potentially autonomous, interaction in asset-twin coupled dynamical systems. In particular, they allow for reasoning and decision-making under uncertainty by blending data assimilation and inference via DL models (from physical to digital) with prediction, planning, and control (from digital to physical). These models form the foundation for the DT framework proposed in Chapter 8.

This page intentionally left blank.

5 DEEP METRIC LEARNING FOR DAMAGE LOCALIZATION

Part of the research presented in this Chapter has been published as

[164] *Torzoni et al., Structural health monitoring of civil structures: A diagnostic framework powered by deep metric learning,* Computers & Structures, *Vol. 271, 2022, p. 106858.*

[167] *Torzoni et al., 5 Damage identification using physics-based datasets: From convolutional to metric-informed damage-sensitive feature extractors,* Data-Centric Structural Health Monitoring: Mechanical, Aerospace and Complex Infrastructure Systems, De Gruyter, *2023, pp. 101–124.*

This Chapter is devoted to the development of a data-driven diagnostic framework for real-time damage localization, which can also be extended to address the identification of operational conditions or the task of damage quantification, as shown in Sec. 6.3 and in Sec. 7.2, respectively. The method is rather general, making it easily adaptable to handle data of different natures, or to solve inverse problems in other application domains.

We integrate aspects from both metric learning and dimensionality reduction techniques. *Metric learning* aims to learn a *distance function* that conforms to a task-specific definition of *similarity*, ensuring that neighbors of a data point are mapped closer than non-neighbors in the learned metric space [151,152,153]. Since it is often challenging to directly infer quantities of interest from sensors data, due to the curse of dimensionality and the low-sensitivity of raw data to damage, *dimensionality reduction* is employed to map high-dimensional data onto an informative low-dimensional manifold. Specifically, we leverage a learnable nonlinear mapping of raw vibration measurements onto a feature space, where structural health parameters describing damage locations can be easily discriminated. This mapping is devised by means of a DNN and learned under the constraint of inducing a damage position-related metric in the low-dimensional space. To achieve this, the training stage is performed in a *supervised pairwise fashion*, exploiting a *Siamese* convolutional architecture [156]. When processing previously unseen data, where damage is assumed to be detected but not yet localized, real-time damage localization is achieved by applying a k-NN regressor in the learned low-dimensional feature space.

151. Kaya *et al., Deep Metric Learning: A Survey.* 2019

152. Bellet *et al., A Survey on Metric Learning for Feature Vectors and Structured Data.* 2013

153. Cakir *et al., Deep Metric Learning to Rank.* 2019

156. Bromley *et al., Signature Verification using a "Siamese" Time Delay Neural Network.* 1993

A training dataset of synthetic vibration recordings is generated by simulating the structural response for potential damage scenarios and operational conditions. To alleviate the associated computational burden, the FOM of the monitored structure is replaced with a more computationally efficient yet accurate ROM relying on the reduced basis method.

The effectiveness of the proposed approach lies in its supervised pairwise training, which enables a *damage-sensitive*

manifold. As a regression approach to damage localization, this strategy eliminates the need to define a large number of target classes in advance. Additionally, it avoids the fine-tuning of highly sensitive hyperparameters, such as distance thresholds and decision boundaries, thereby reducing the dependence on user-specific choices.

The proposed methodology is investigated by adopting three alternative loss functions to train the Siamese network and varying the size of the low-dimensional feature space. Although experimental data are not employed, the method is assessed using noise-contaminated synthetic measurements related to an L-shaped cantilever beam and a portal frame railway bridge, demonstrating significant damage localization capabilities.

The Chapter is organized as follows. The deep metric learning framework is detailed in Sec. 5.1, focusing on the data pairing process required for training in Sec. 5.1.1, the characteristics of the employed Siamese architecture in Sec. 5.1.2, and the pairwise loss functions to be minimized in Sec. 5.1.3. The use of a k-NN regressor in the learned low-dimensional space for assessing the location of damage is described in Sec. 5.2. The application of the methodology to an L-shaped cantilever beam and a portal frame railway bridge is considered in Sec. 5.3 and in Sec. 5.4, respectively. Sec. 5.5 finally summarizes the obtained results and draws the conclusions.

5.1 DEEP METRIC LEARNING

In the considered framework, the problem of damage localization is addressed through the construction (learning) of a smooth mapping of labeled data onto a low-dimensional feature space. This mapping must encode the coordinates of a damaged region in a regression-like fashion. Accordingly, the composition of our training dataset is similar to that employed for regression purposes in Eq. (4.1), and is defined as:

$$\mathcal{D}_{\text{LOC}} = \{(\mathbf{U}_i, \mathbf{y}_i)\}_{i=1}^{N_{\text{train}}}. \tag{5.1}$$

In Eq. (5.1), $\mathbf{U}_i(\boldsymbol{\mu}_i) = [\mathbf{u}_1(\boldsymbol{\mu}_i), \dots, \mathbf{u}_{N_u}(\boldsymbol{\mu}_i)]$, $i = 1, \dots, N_{\text{train}}$, is a multivariate time series made of N_u synthetic vibration recordings with L measurements equally spaced in time. Each recording cover a monitoring window $(0, T)$, short enough for the operational and damage conditions to be considered time-invariant. To populate $\mathcal{D}_{\text{LOC}}$, the input parameters $\boldsymbol{\mu}$ are taken as uniformly distributed and sampled using the Latin hypercube rule, as described in Sec. 3.2. The structural response for the i-th sampling of input parameters $\boldsymbol{\mu}_i$ is then simulated following

the procedure detailed in Sec. 3.3.2. Sensor recordings are extracted using Eq. (3.23), corrupted by adding an independent, identically distributed Gaussian noise, and labeled with the corresponding spatial coordinates $\mathbf{y}_i$ of the applied stiffness reduction. As described in Sec. 3.2, $\mathbf{y}_i$ identifies the coordinates of the center of mass of a subdomain $\Omega_{\mathbf{y}}$, where damage is simulated by reducing the material stiffness by a magnitude δ_i. While each instance is characterized by a stiffness reduction of different magnitude to account for uncertainties related to its value, this variation is not part of the label, as the focus is only on the damage localization task. In the following, the index i will be omitted for ease of notation, unless necessary.

The FOMs and ROMs have been implemented in the `Matlab` environment, using the `redbKIT` library [330]. All computations have been carried out on a personal computer featuring an `AMD Ryzen™ 9 5950X` CPU @ 3.4 GHz and 128 GB RAM.

330. Negri, *redbKIT, version 2.2.* 2016

For any given distance measure, a feature extraction algorithm can be viewed as a means to learn a *similarity metric* among the input data [331]. In this context, a *regressor* $\mathrm{NN}_{\mathbf{h}}$ is designed to approximate a target function $\mathrm{NN}_{\mathbf{h}} : \mathbb{R}^{L \times N_u} \mapsto \mathbb{R}^{N_h}$, $\mathbf{U} \mapsto \mathbf{h}$, mapping the vibration recordings $\mathbf{U}$ onto their feature representation $\mathbf{h}$ in a low-dimensional space of size N_h.

331. Salakhutdinov *et al.*, *Learning a Nonlinear Embedding by Preserving Class Neighborhood Structure.* 2007

The purpose of $\mathrm{NN}_{\mathbf{h}}$ is to enable the computation of a distance function $\mathcal{E}_{\mathbf{h}} = \mathcal{E}_{\mathbf{h}}(\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1), \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2))$, according to a chosen metric (e.g., cosine or Euclidean), between any pair of mappings $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1)$ and $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)$. In particular, $\mathcal{E}_{\mathbf{h}}$ is required to approximate, at least semantically, the Euclidean distance $\mathcal{E}_{\mathbf{y}} = \|\mathbf{y}_1 - \mathbf{y}_2\|_2$ between the associated target labels $\mathbf{y}_1$ and $\mathbf{y}_2$.

5.1.1 Data pairing process

A possible approach to learn $\mathrm{NN}_{\mathbf{h}}$ relies on *pairwise learning*, which implies that training instances are processed in pairs. In this context, the training dataset $\mathcal{D}_{\mathrm{LOC}}$ is augmented through a *data pairing process*. By fixing a threshold distance $\overline{\mathcal{E}_{\mathbf{y}}} \in \mathbb{R}$ to characterize the similarity of data in terms of $\mathbf{y}$, for each instance $(\mathbf{U}_i, \mathbf{y}_i)$ in $\mathcal{D}_{\mathrm{LOC}}$, we assemble $N_+ \in \mathbb{N}$ *positive pairs of similar instances* characterized by $\mathcal{E}_{\mathbf{y}} \leq \overline{\mathcal{E}_{\mathbf{y}}}$, and $N_- \in \mathbb{N}$ *negative pairs of dissimilar instances* characterized by $\mathcal{E}_{\mathbf{y}} > \overline{\mathcal{E}_{\mathbf{y}}}$, as follows:

$$\mathcal{D}^{\mathrm{P}}_{\mathrm{LOC}} = \{(\mathbf{U}_1, \mathbf{y}_1, \mathbf{U}_2, \mathbf{y}_2)_\tau\}_{\tau=1}^{N^{\mathrm{P}}_{\mathrm{train}}}, \tag{5.2}$$

where $N^{\mathrm{P}}_{\mathrm{train}} = N_{\mathrm{train}}(N_+ + N_-)$ is the total number of pairs.

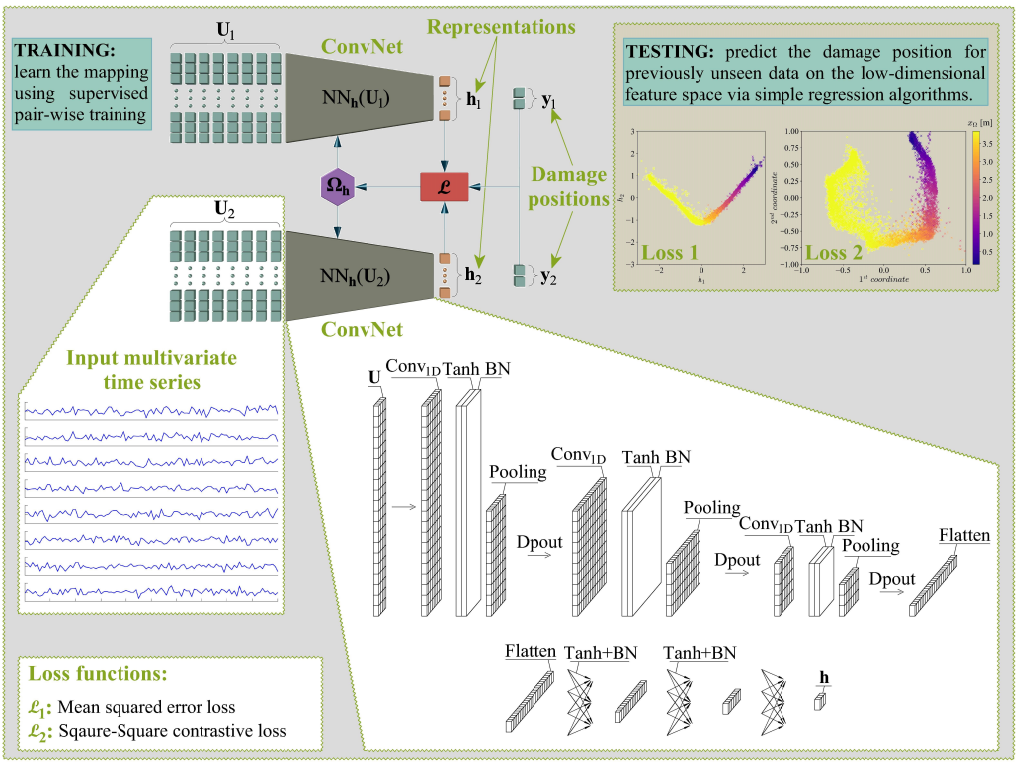


Figure 5.1: Deep metric learning damage localization framework. The sub-network $\mathrm{NN}_{\mathbf{h}}$ takes vibration recordings $\mathbf{U}$ and provides the corresponding feature representation $\mathbf{h}$ in a low-dimensional space. The weights $\mathbf{\Omega}_{\mathbf{h}}$ parametrizing $\mathrm{NN}_{\mathbf{h}}$ are learned by processing paired vibration recordings $\{\mathbf{U}_1, \mathbf{U}_2\}$, and minimizing a functional $\mathcal{L}$ of the distance $\mathcal{E}_{\mathbf{h}}$ between two low-dimensional representations $\{\mathbf{h}_1, \mathbf{h}_2\}$ and of the Euclidean distance $\mathcal{E}_{\mathbf{y}}$ between the associated labels $\{\mathbf{y}_1, \mathbf{y}_2\}$. For data unseen during training, the damage position is predicted applying a simple regression algorithm in the learned feature space.

5.1.2 SIAMESE ARCHITECTURE

The Siamese architecture, originally proposed for signature verification [156], allows for pairwise learning and can be thought of as a network made of two twins $\mathrm{NN}_{\mathbf{h}}$ linked by a loss function $\mathcal{L}$, see Fig. 5.1. Here, the two sub-networks share the same set of parameters $\mathbf{\Omega}_{\mathbf{h}}$, and each of them is therefore simply referred to as the *sub-network* $\mathrm{NN}_{\mathbf{h}}$. During training, paired vibration recordings $\mathbf{U}_1$ and $\mathbf{U}_2$ are jointly processed as $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1)$ and $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)$, so that the associated distance $\mathcal{E}_{\mathbf{h}}(\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1), \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2))$ can be computed to evaluate the loss function $\mathcal{L}$ ruling the learning of $\mathbf{\Omega}_{\mathbf{h}}$, as detailed in Sec. 5.1.3.

The mapping provided by $\mathrm{NN}_{\mathbf{h}}$ is required to be invariant under transformations on the input space not related to the damage position. To achieve this goal, a *damage-sensitive feature extractor* is designed at the root of $\mathrm{NN}_{\mathbf{h}}$ through the composition of three *1D CNN blocks*. Each convolutional block consists of a 1D convolutional layer, a Tanh activation function, a batch normalization layer, a 1D max pooling layer, and a dropout layer for regularization. The extracted features are expected to be sensitive to the presence of damage, while remaining insensitive to measurement noise and operational variability. The output features are then flattened and run through a stack of three

dense layers providing the low-dimensional representation $\mathbf{h}$. A schematic of the sub-network $\mathrm{NN}_{\mathbf{h}}$ is outlined in Fig. 5.1.

It is worth noting that the modeling choices for $\mathrm{NN}_{\mathbf{h}}$ are suited to the specific characteristics of the considered observational data. However, the overall framework is rather general, admitting different modeling choices tailored to the data and the characteristics of the problem at hand. $\mathrm{NN}_{\mathbf{h}}$ has been implemented through the `Tensorflow`-based `Keras` API [250], and trained on a single `Nvidia GeForce RTX™ 3080` GPU card. The relevant implementation details are reported in Appendix A.

250. Chollet *et al.*, *Keras*. 2015

5.1.3 PAIRWISE LOSS FUNCTIONS

The set of learnable weights $\mathbf{\Omega_h}$ parametrizing $\mathrm{NN}_{\mathbf{h}}$ is optimized, considering the minimization of three alternative loss functions:

$$\mathcal{L}_{\mathbf{h}}^{(1)}(\mathbf{\Omega_h}, \mathcal{D}_{\mathrm{LOC}}^{\mathrm{P}}) = \frac{1}{N_{\mathrm{train}}^{\mathrm{P}}} \sum_{\tau=1}^{N_{\mathrm{train}}^{\mathrm{P}}} (\mathcal{E}_{\mathbf{y}} - \mathcal{E}_{\mathbf{h}})_{\tau}^{2}, \tag{5.3}$$

$$\mathcal{L}_{\mathbf{h}}^{(2,3)}(\mathbf{\Omega_h}, \mathcal{D}_{\mathrm{LOC}}^{\mathrm{P}}) = \frac{1}{N_{\mathrm{train}}^{\mathrm{P}}} \sum_{\tau=1}^{N_{\mathrm{train}}^{\mathrm{P}}} \left\{ \frac{1-\Gamma}{2} \mathcal{E}_{\mathbf{h}}^{2} + \frac{\Gamma}{2} \left[\max\left(0, \overline{\epsilon_{\mathbf{h}}} - \mathcal{E}_{\mathbf{h}}\right)\right]^{2} \right\}_{\tau}, \tag{5.4}$$

with the difference between $\mathcal{L}_{\mathbf{h}}^{(2)}$ and $\mathcal{L}_{\mathbf{h}}^{(3)}$ standing in the choice of the associated distance $\mathcal{E}_{\mathbf{h}}$ — this is clarified in the following. In particular, $\mathcal{L}_{\mathbf{h}}^{(1)}$ is the MSE between $\mathcal{E}_{\mathbf{h}}$ and $\mathcal{E}_{\mathbf{y}}$, proving a measure of the distance between the target distance $\mathcal{E}_{\mathbf{y}}$ and the approximated distance $\mathcal{E}_{\mathbf{h}}$ from $\mathrm{NN}_{\mathbf{h}}$. On the other hand, $\mathcal{L}_{\mathbf{h}}^{(2,3)}$ is the *pairwise contrastive loss* [154,155], or *Square-Square loss* [332], useful to learn a geometric structure characterizing the low-dimensional feature space. Herein: $\Gamma \in \{0, 1\}$, where $\Gamma = 0$ indicates that $\mathbf{y}_1$ and $\mathbf{y}_2$ identify a positive pair, and $\Gamma = 1$ indicates a negative pair; and $\overline{\epsilon_{\mathbf{h}}} > 0$ is a margin beyond which negative pairs do not contribute to $\mathcal{L}_{\mathbf{h}}^{(2,3)}$. The minimization of $\mathcal{L}_{\mathbf{h}}^{(2,3)}$ yields a distance function $\mathcal{E}_{\mathbf{h}}$, such that negative pairs are kept away by at least the margin $\overline{\epsilon_{\mathbf{h}}}$, while positive pairs are pushed to be as close as possible. In the case of $\mathcal{L}_{\mathbf{h}}^{(2,3)}$, label information implicitly guide the dimensionality reduction to encode sensitivity to $\mathbf{y}$, thereby enabling a manifold to suitably describe the parametric space underlying the processed measurements.

154. Hadsell *et al.*, *Dimensionality Reduction by Learning an Invariant Mapping*. 2006

155. Chopra *et al.*, *Learning a similarity metric discriminatively, with application to face verification*. 2005

332. LeCun *et al.*, *A tutorial on energy-based learning*. 2006

The metric employed for the computation of $\mathcal{E}_{\mathbf{h}}$ for $\mathcal{L}_{\mathbf{h}}^{(1)}$ and $\mathcal{L}_{\mathbf{h}}^{(2)}$ is the Euclidean one, while $\mathcal{L}_{\mathbf{h}}^{(3)}$ exploits the *cosine similarity*

$C_s \in \mathbb{R}$ as the relevant metric. In detail:

$$\mathcal{E}_{\mathbf{h}} = \begin{cases} \|\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1) - \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)\|_2 & \text{for } \mathcal{L}_1,\ \mathcal{L}_2, \\ 1 - C_s(\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1), \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)) & \text{for } \mathcal{L}_3, \end{cases} \tag{5.5}$$

where the cosine similarity is given as:

$$C_s(\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1), \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)) = \frac{\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1) \cdot \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)}{\|\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1)\|_2 \|\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)\|_2}, \tag{5.6}$$

measuring the cosine of the angle between $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1)$ and $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)$. If $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_1)$ and $\mathrm{NN}_{\mathbf{h}}(\mathbf{U}_2)$ are parallel, $C_s = 1$; if they are orthogonal, $C_s = 0$. The reason to consider $\mathcal{L}_{\mathbf{h}}^{(3)}$ as an additional loss function is that, employing the cosine similarity as the metric for $\mathcal{E}_{\mathbf{h}}$, *embeddings* associated with negative pairs are constrained to feature a relative rotation in the low-dimensional space, rather than a different module. This is expected to induce an auto-regularization effect that stabilizes the training and improves the representation capability of $\mathrm{NN}_{\mathbf{h}}$.

For any given set of input vibration recordings $\mathbf{U}$, the trained $\mathrm{NN}_{\mathbf{h}}$ provides the associated feature representation $\mathbf{h}$, as follows:

$$\mathbf{h} = \mathrm{NN}_{\mathbf{h}}(\mathbf{U}). \tag{5.7}$$

5.1.4 Energy-based models perspective

An interesting perspective to frame the proposed strategy can be built upon *energy-based models* [332]. Energy-based models rely on an *energy function* $\Lambda : (\mathbb{R}^{N_{\boldsymbol{x}}}, \mathbb{R}^{N_{\boldsymbol{y}}}) \mapsto \mathbb{R}$, $(\boldsymbol{x}, \boldsymbol{y}) \mapsto \mathbb{R}$, that measures the level of compatibility between $N_{\boldsymbol{x}} \in \mathbb{N}$ observed variables $\boldsymbol{x}$ and $N_{\boldsymbol{y}} \in \mathbb{N}$ variables $\boldsymbol{y}$ to be inferred. In our case, $\boldsymbol{x}$ represents the observed vibration recordings, and $\boldsymbol{y}$ represents potential damage positions. The learning phase consists in finding, among a family of energy functions $\Lambda_{\boldsymbol{\Omega}}(\boldsymbol{x}, \boldsymbol{y})$ parametrized by tunable weights $\boldsymbol{\Omega}$, the one that provides low energy values for a correct association of $\boldsymbol{y}$ to $\boldsymbol{x}$ and high energy values otherwise. $\Lambda_{\boldsymbol{\Omega}}(\boldsymbol{x}, \boldsymbol{y})$ is shaped through the minimization of a loss functional $\mathcal{L}(\mathcal{D}, \Lambda_{\boldsymbol{\Omega}}(\boldsymbol{x}, \boldsymbol{y}))$ while optimizing $\boldsymbol{\Omega}$ on a dataset $\mathcal{D}$. The inference of $\boldsymbol{y}$ thus consists in simply picking its value that minimizes the learned energy function $\Lambda_{\boldsymbol{\Omega}}$, given $\boldsymbol{x}$.

332. LeCun *et al.*, *A tutorial on energy-based learning*. 2006

Contrastive methods shape the energy function through a loss functional designed to yield low values for either low energy on correct associations or high energy on incorrect associations, and high values for either high energy on correct associations or low energy on incorrect associations. In our case, the energy function of interest is represented by $\mathcal{E}_{\mathbf{h}}$, and $\mathcal{L}_{\mathbf{h}}^{(1,2,3)}$ are the considered loss functionals. While $\mathcal{L}_{\mathbf{h}}^{(1)}$ implicitly pushes up the

energy for all the possible outcomes, pushing down the energy for the correct association, $\mathcal{L}_{\mathbf{h}}^{(2)}$ and $\mathcal{L}_{\mathbf{h}}^{(3)}$ instead push down the energy for the correct association and push up the energy above a prescribed margin for incorrect answers.

5.2 Regression in the Low-Dimensional Metric Space

Once $\mathrm{NN}_{\mathbf{h}}$ is trained, the damage position for unseen query data is predicted through a *k-NN regressor* $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}^3$, $\mathbf{h} \mapsto \mathbf{y}$, applied on top of mapping $\mathrm{NN}_{\mathbf{h}}$, similarly to [333]. The k-NN algorithm is chosen not only for its effectiveness but also because it is a nonparametric, memory-based method, making it exceptionally fast in both the learning and prediction stages, especially when dealing with limited data dimensionality.

333. Liu *et al.*, *Metric-Based Semi-Supervised Regression*. 2020

To fit $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$, the Siamese architecture as well as $\mathcal{D}_{\text{LOC}}^{\text{P}}$ are discarded, retaining only the trained sub-network $\mathrm{NN}_{\mathbf{h}}$ and $\mathcal{D}_{\text{LOC}}$. Subsequently, $\mathcal{D}_{\text{LOC}}$ is pre-processed using $\mathrm{NN}_{\mathbf{h}}$ to generate a dedicated training dataset to feed $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$, similar to that in Eq. (4.2). By mapping the vibration recordings in $\mathcal{D}_{\text{LOC}}$ onto the learned feature space, once and for all, this training dataset is obtained as follows:

$$\mathcal{D}_{\text{LOC}}^{\mathbf{h}} = \{(\mathbf{h}_i = \mathrm{NN}_{\mathbf{h}}(\mathbf{U}_i), \mathbf{y}_i)\}_{i=1}^{N_{\text{train}}}, \tag{5.8}$$

so that the low-dimensional features $\mathbf{h}_i$ and the corresponding labels $\mathbf{y}_i$, with $i = 1, \ldots, N_{\text{train}}$, can be stored by $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$. The number N_k of nearest neighbors considered in the k-NN regression and the associated weighting rule are determined through a cross-validation strategy, as described in Sec. 2.2.

It might be noticed that fitting $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$ on the reference low-dimensional representations $\mathbf{h}_i$, $i = 1, \ldots, N_{\text{train}}$, may result in limited generalization capabilities. Typically, data-driven algorithms best perform on training data, thus the low-dimensional representation associated with the data used to train $\mathrm{NN}_{\mathbf{h}}$ may be too optimistic compared to those obtained for unseen data. Although in our experiments this issue does not show up due to the effective prevention of overfitting, leveraging an additional set of recordings to fine-tune $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$ could be beneficial for enhanced damage localization.

The algorithmic description of the preliminary offline phase of the proposed strategy is reported in Algorithm 5.1. During the next online phase, unseen incoming data, for which damage is assumed to be detected but not yet localized, are processed to provide real-time damage localization as described below.

Algorithm 5.1 Deep Metric Learning for Damage Localization. Preliminary offline phase: algorithmic description

Input: parametrization of the physics-based model
POD tolerance ϵ_σ
training options for the feature extractor $\mathrm{NN}_{\mathbf{h}}$
training options for the k-NN regressor $\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$

set up the numerical model of the structure
assemble the snapshot matrix $\mathbf{S}$ via FOM analyses
compute the POD basis matrix via SVD of $\mathbf{S}$
use the ROM to populate the training dataset $\mathcal{D}_{\text{LOC}}$
augment $\mathcal{D}_{\text{LOC}}$ to $\mathcal{D}^{\text{P}}_{\text{LOC}}$ through the pairing process
train $\mathrm{NN}_{\mathbf{h}}$ (Siamese network) on $\mathcal{D}^{\text{P}}_{\text{LOC}}$
use the trained $\mathrm{NN}_{\mathbf{h}}$ to derive $\mathcal{D}^{\mathbf{h}}_{\text{LOC}}$ from $\mathcal{D}_{\text{LOC}}$
fit $\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$ on $\mathcal{D}^{\mathbf{h}}_{\text{LOC}}$ via cross-validation
return trained feature extractor $\mathrm{NN}_{\mathbf{h}}$,
trained k-NN regressor $\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$

Let $\mathbf{U}^{\text{EXP}}$ denote a query instance of vibration recordings, and $\mathbf{h}^{\text{EXP}} = \mathrm{NN}_{\mathbf{h}}(\mathbf{U}^{\text{EXP}})$ represent the corresponding query low-dimensional features. Then, $\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$ computes the distances between $\mathbf{h}^{\text{EXP}}$ and its N_k nearest neighbors among $\mathbf{h}_i$, $i = 1, \ldots, N_{\text{train}}$. Subsequently, it estimates the associated damage position as a weighted average of the N_k nearest neighbors labels $\widehat{\mathbf{y}} = \mathcal{M}^{\mathbf{y}}_{\text{k-NN}}(\mathbf{h}^{\text{EXP}})$, following Eq. (2.15).

5.3 Damage localization: L-shaped cantilever beam

The first case study used to assess the effectiveness of the proposed methodology involves the L-shaped cantilever beam depicted in Fig. 5.2. The structure consists of two arms, each with dimensions of 4 m in length, 0.3 m in width, and 0.4 m in height. The assumed mechanical properties are those of concrete: Young's modulus $E = 30$ GPa, Poisson's ratio $\nu = 0.2$, density $\rho = 2500$ kg/m^3. The structure is subjected to a distributed vertical load $q(t)$, applied to an area of (0.3×0.3) m^2 near to its tip. The load varies in time according to:

$$q(t) = Q \sin(2\pi f_q t), \tag{5.9}$$

where $Q \in [1, 5]$ kPa and $f_q \in [10, 100]$ Hz represent the load amplitude and frequency, respectively. Consistent with the setup described in Sec. 3.2, these parameters follow a uniform distribution within their respective ranges.

The goal of this first test case is to assess the damage localization capability of the proposed method while highlighting the

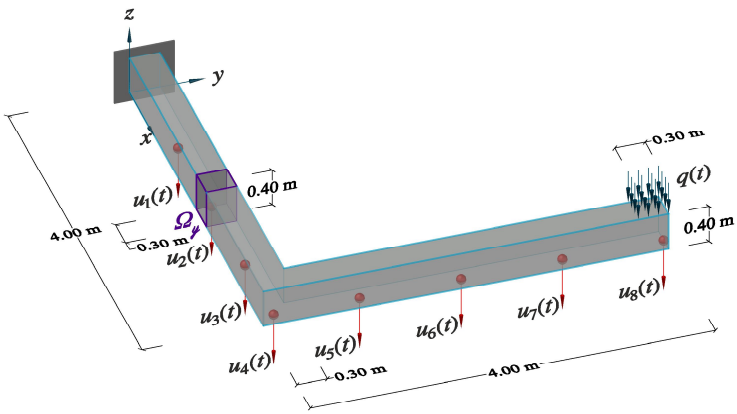


Figure 5.2: L-shaped cantilever beam: details of synthetic recordings related to displacements $u_1(t), \ldots, u_8(t)$, loading condition, and damageable region $\Omega_{\mathbf{y}}$.

interpretability of the results. Additionally, this test case aims to determine the optimal combination of the loss function and the size N_h of the feature space that yields the best results.

5.3.1 Dataset assembly and training options

Synthetic displacement time histories $\mathbf{U}(\boldsymbol{\mu}) = [\mathbf{u}_1(\boldsymbol{\mu}), \ldots, \mathbf{u}_{N_u}(\boldsymbol{\mu})]$ are generated in relation to $N_u = 8$ DOFs along the bottom surface of the structure, to mimic the monitoring system depicted in Fig. 5.2. Each recording spans a time interval $(0, T = 1\text{ s})$ with an acquisition frequency of $f_s = 500$ Hz. To emulate sensors self-noise, the recordings are contaminated with additive Gaussian noise to yield a *signal-to-noise ratio* (SNR) of 80*.

*Throughout this Thesis, we define the signal-to-noise ratio as the ratio of the variance of the (meaningful) signal to the variance of the noise, assuming zero-mean noise

Damage is simulated within a subdomain $\Omega_{\mathbf{y}}$ of dimensions $(0.3 \times 0.3 \times 0.4)$ m^3. The location of $\Omega_{\mathbf{y}}$ is parametrized through $\mathbf{y} = (x_\Omega, y_\Omega)^\top$, with either x_Ω or y_Ω varying in the range $[0.15, 3.85]$ m — thus, it can be represented as an abscissa running along the axis of the structure. The damage level δ can take values within the range $\delta \in [10\%, 25\%]$, and it remains constant within the time interval $(0, T)$.

The FOM is established through a finite element discretization using linear tetrahedral elements, resulting in $N_{\text{FE}} = 4659$ DOFs. Due to its minimal relevance in identifying continuously excited systems, as indicated in [57,58], structural damping is neglected. The basis matrix $\mathbf{W}$ underlying the ROM is derived from a snapshot matrix $\mathbf{S}$, assembled by evaluating the FOM over $(0, T)$, for $N_P = 300$ varied values of the input parameters $\boldsymbol{\mu} = (Q, f_q, \mathbf{y}, \delta)^\top$ sampled via the Latin hypercube rule. By prescribing a tolerance $\epsilon_\sigma = 0.0005$ on the fraction of energy content to be neglected in the approximation, the order of the ROM turns out to be $N_{\text{RB}} = 68$. The computing time required by each simulation accordingly decreases from 11.30 s to 0.39 s, entailing a speed-up of approximately 29 times.

57. Eftekhar Azam *et al.*, *Online damage detection via a synergy of proper orthogonal decomposition and recursive Bayesian filters*. 2017

58. Corigliano *et al.*, *Parameter identification in explicit structural dynamics: Performance of the extended Kalman filter*. 2004

For the case at hand, the training dataset $\mathcal{D}_{\text{LOC}}$ is populated with $N_{\text{train}} = 8000$ instances collected through the ROM. This dataset is then employed to assemble its augmented version $\mathcal{D}^{\text{P}}_{\text{LOC}}$, according to Eq. (5.2). To this aim, we set a threshold distance $\mathcal{E}_{\mathbf{y}} = 0.15$ m based on the size of the damaged subdomain $\Omega_{\mathbf{y}}$. Each instance in $\mathcal{D}_{\text{LOC}}$ is thus paired with $N_{+} = 10$ similar instances, characterized by $\mathcal{E}_{\mathbf{y}} \leq \overline{\mathcal{E}_{\mathbf{y}}}$, and with $N_{-} = 10$ dissimilar instances, characterized by $\mathcal{E}_{\mathbf{y}} > \overline{\mathcal{E}_{\mathbf{y}}}$.

Due to the lack of experimental data, the assessment of the damage localization capabilities enabled by $\text{NN}_{\mathbf{h}}$ is carried out on a testing set comprising $N_{\text{test}} = 800$ *pseudo-experimental instances*, derived from noise-corrupted FOM solutions.

5.3.2 Localization outcome: numerical results

We now present the damage localization outcomes obtained by training $\text{NN}_{\mathbf{h}}$ using the three alternative loss functions $\mathcal{L}^{(1)}_{\mathbf{h}}$, $\mathcal{L}^{(2)}_{\mathbf{h}}$, and $\mathcal{L}^{(3)}_{\mathbf{h}}$ introduced in Sec. 5.1.3. The analysis is systematically repeated for various sizes N_h of the low-dimensional space, to provide insights into the selection of its optimal value.

Loss function $\mathcal{L}^{(1)}_{\mathbf{h}}$. We begin by presenting the results obtained using the MSE-based loss function $\mathcal{L}^{(1)}_{\mathbf{h}}$. In this case, the choice of the low-dimensional space size $N_h = 2$ directly follows from the modeling of the position of $\Omega_{\mathbf{y}}$, requiring no additional tuning. The low-dimensional embeddings $\{\mathbf{h}_i = (h_1, h_2)^{\top}_i\}_{i=1}^{N_{\text{train}}}$ provided by $\text{NN}_{\mathbf{h}}$ for the training data are illustrated in Fig. 5.3a. The two scatter plots include a color channel referring to the target damage position along the x-axis, denoted as $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$, and along the y-axis, denoted as $\{y_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ — refer also to Fig. 5.2. We observe how the overall shape defined by the scatter plots aligns with the structural layout, and the colors mostly correspond to the coordinates in the physical space. As desired, situations undergoing a similar damage condition, in terms of damage position, are mapped close in the learned space. This visual check offers an initial qualitative indication of the effectiveness of the learned mapping $\text{NN}_{\mathbf{h}}$, and suggests that the low-dimensional representation provides a coordinate system for capturing the damage position from vibration recordings.

The testing phase for $\text{NN}_{\mathbf{h}}$ aims to mimic a real structural monitoring process. Raw vibration recordings are assumed to be collected on-the-fly and processed to determine the damage position. The embeddings associated with the N_{test} testing instances are illustrated in Fig. 5.3b, demonstrating that $\text{NN}_{\mathbf{h}}$ exhibits proper generalization capabilities to previously unseen

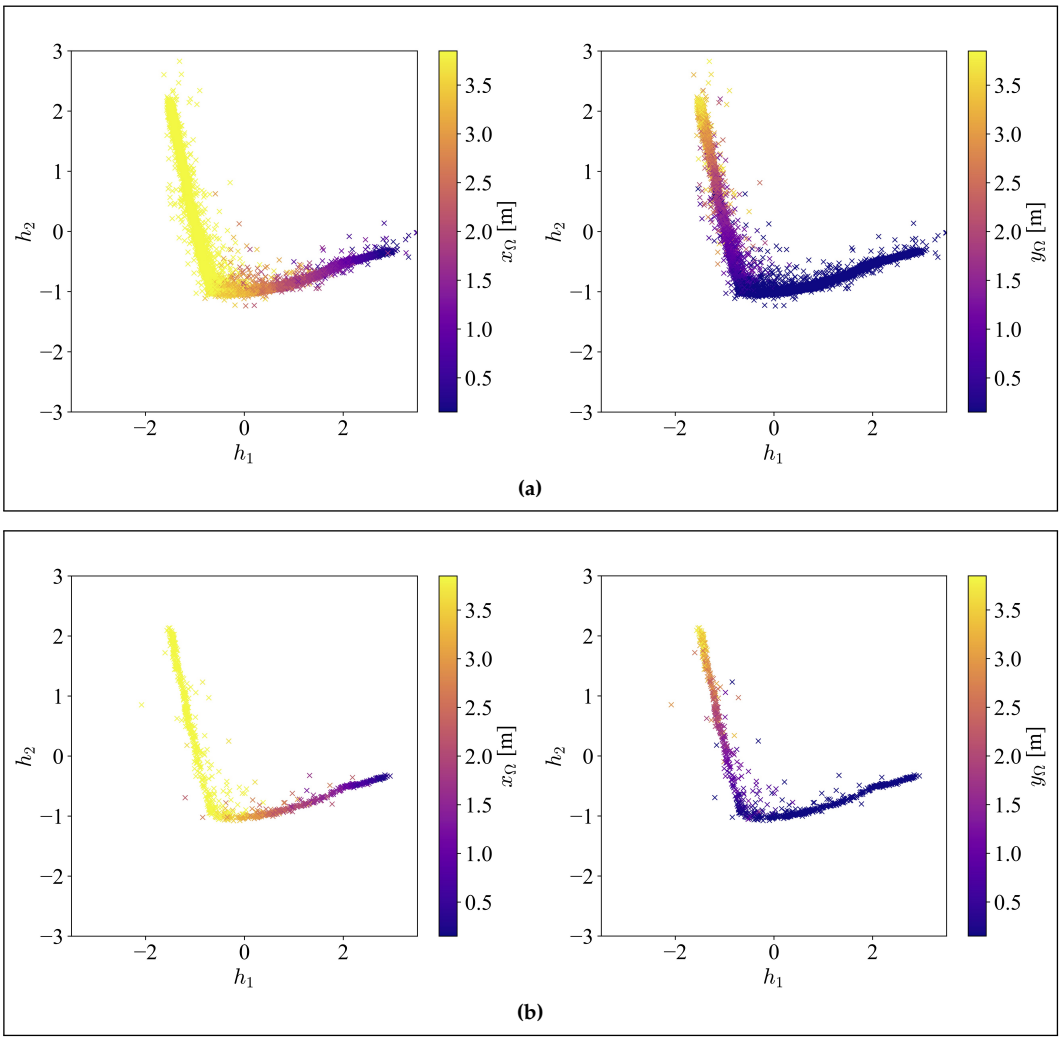


Figure 5.3: L-shaped cantilever beam - Low-dimensional embeddings using $\mathcal{L}_{\mathbf{h}}^{(1)}$: (a) training data representations $\{\mathbf{h}_i = (h_1, h_2)_i^\top\}_{i=1}^{N_{\text{train}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (right) y-direction $\{y_{\Omega i}\}_{i=1}^{N_{\text{train}}}$; (b) testing data representations $\{\mathbf{h}_j = (h_1, h_2)_j^\top\}_{j=1}^{N_{\text{test}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (right) y-direction $\{y_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

data. Real-time damage localization is then achieved via k-NN regression. In the current scenario, $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$ is fitted using a 10-fold cross-validation. The maximum fold-averaged regression score, measured via negative RMSE, is achieved with $N_k = 58$ neighbors and a uniform weighting rule.

The distributions of *prediction errors* along the x and y directions for the testing data are presented in Fig. 5.4. In these charts, the *counts* represent the number of instances for which a particular prediction error is observed. The reported values are distributed within a relatively narrow range around the origin, indicating that the damage position is mostly located with high accuracy. It is worth noting that the error distribution along the y-direction exhibits a slightly greater variance compared to the x-direction. This difference is attributed to the structural layout, where the sensitivity of sensor recordings to damage decreases

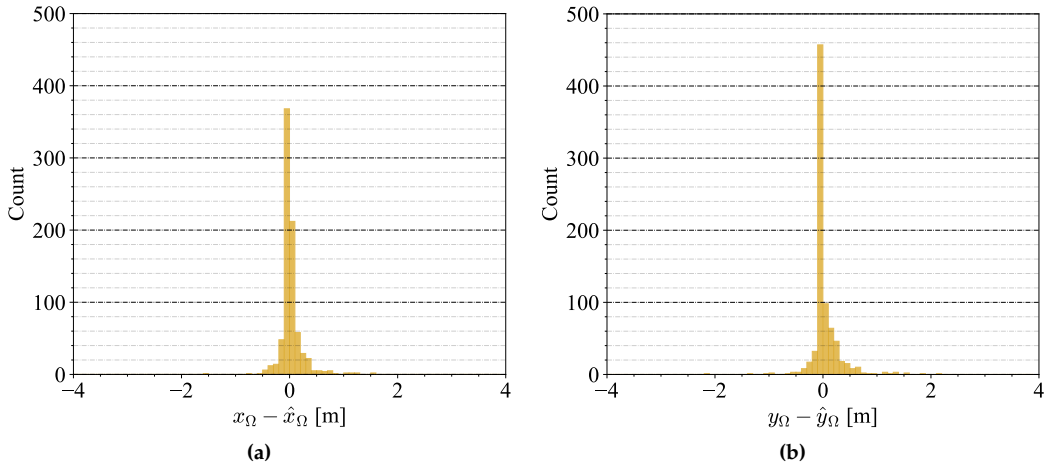


Figure 5.4: L-shaped cantilever beam - Distribution of prediction errors along the (a) x-direction and (b) y-direction, using $\mathcal{L}_{\mathbf{h}}^{(1)}$.

as the damage position moves away from the clamped side.

Assuming that the prediction errors along the x and y directions follow Gaussian distributions, relevant statistics are presented in Tab. 5.1. The metrics used for evaluating the performance of the damage localization model include MAE, error mean, and error standard deviation (StdDev) along both the x and y directions. It is worth noting that the MAE values are smaller than half the size of the damaged region in both directions, indicating high precision. The mean values, close to zero, suggest an unbiased estimator, while the StdDev values confirm the greater inaccuracies along the y-direction, consistent with the MAE values. Based on these statistics, it is expected that 95% of the testing population will yield prediction errors lying within the intervals $[-0.38, 0.43]$ m and $[-0.43, 0.51]$ m, respectively along the x and y directions. This estimation provides further insights into the model's performance and the expected range of prediction errors.

Loss function $\mathcal{L}_{\mathbf{h}}^{(2)}$. We now present the results obtained using the Square-Square loss function $\mathcal{L}_{\mathbf{h}}^{(2)}$ equipped with the Euclidean metric. The Square-Square loss function is designed to keep negative pairs apart by at least the margin $\overline{\epsilon_{\mathbf{h}}}$, while pushing positive pairs to be as close as possible. Unlike for $\mathcal{L}_{\mathbf{h}}^{(1)}$, which explicitly constraints the mapping under the guidance of the labels, $\mathrm{NN}_{\mathbf{h}}$ is left free to learn a suitable metric space by itself. This aspect results in the need of increasing the size N_h of the low-dimensional space to provide $\mathrm{NN}_{\mathbf{h}}$ with a sufficient representation capability to properly describe the variability of the monitored system in terms of structural health.

In Sec. 4.2.7, we have discussed how determining an optimal

N_k:	58	Weighting rule:	Uniform
MAE($x_\Omega, \widehat{x}_\Omega$):	0.094 m	MAE($y_\Omega, \widehat{y}_\Omega$):	0.117 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.028 m	Mean($y_\Omega - \widehat{y}_\Omega$):	0.043 m
StdDev($x_\Omega - \widehat{x}_\Omega$):	0.21 m	StdDev($y_\Omega - \widehat{y}_\Omega$):	0.24 m

Table 5.1: L-shaped cantilever beam - Accuracy of damage localization along the x and y directions using $\mathcal{L}_{\mathbf{h}}^{(1)}$.

size for the latent space of an AE remains an open question. The same consideration applies to the size N_h of the low-dimensional space described by $\text{NN}_{\mathbf{h}}$. We argue that a redundant encoding may help in handling the ill-posedness of the sought mapping. To identify a suitable N_h value, the damage localization capabilities enabled by $\text{NN}_{\mathbf{h}}$ are systematically analyzed for $N_h = \{2, 3, 4\}$. As we will see, the best outcome is obtained with $N_h = 4$, with no significant improvements for $N_h > 4$. This result is somehow expected, as $N_h = 4$ matches the number of independent parameters responsible for the data variability, i.e., the *intrinsic dimension* of the structural model.

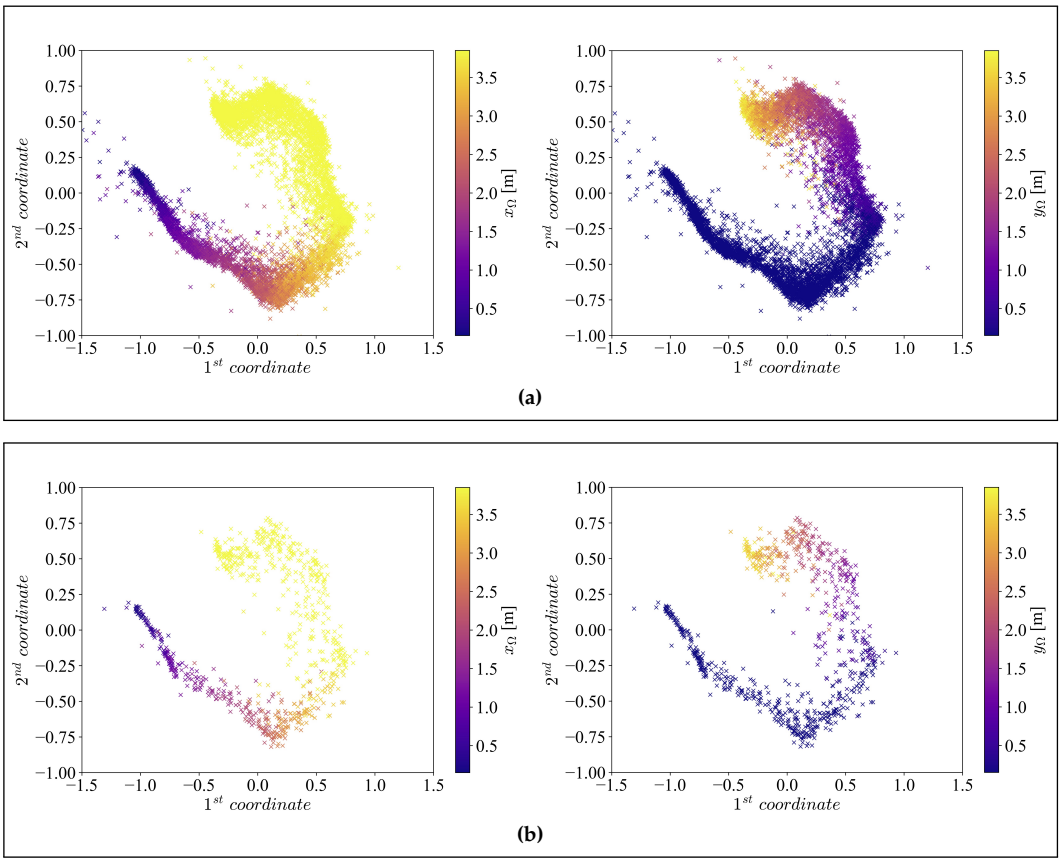
A compact representation of the low-dimensional embeddings provided by $\text{NN}_{\mathbf{h}}$ for the training and testing data in the case of $N_h = 4$ is reported in Fig. 5.5. These scatter plots depict a downsized version of the extracted low-dimensional features, obtained by means of *multidimensional scaling* (MDS). MDS is a nonlinear dimensionality reduction method in statistical learning that seeks a low-dimensional representation of the input data while preserving pairwise scalar products as much as possible [8,334]. We employ the *metric* MDS implementation in `scikit-learn` [335]. As expected, the overall shape defined by the scatter plots does not resemble the actual structural layout in this case, as the relevant geometrical information is not explicitly exploited during training. However, we note how the resulting manifold effectively encodes the sensitivity of the processed measurements to the position of damage. Additional scatter plots displaying the source low-dimensional embeddings of the training and testing data are presented in Fig. 5.6 and Fig. 5.7, respectively. Here, the low-dimensional features $\{\mathbf{h}_i = (h_1, h_2, h_3, h_4)_i^\top\}_{i=1}^{N_{\text{train}}}$ and $\{\mathbf{h}_j = (h_1, h_2, h_3, h_4)_j^\top\}_{j=1}^{N_{\text{test}}}$ are plotted showing two components at a time, against the correct damage position along the x and y directions. Both the MDS representations and the source low-dimensional embeddings clearly show that the task-specific mapping is learned with proper generalization capabilities to previously unseen data, as consistent results are obtained for the training and testing data.

The distributions of the prediction errors along the x and y directions for the testing data are reported in Fig. 5.8, focusing on the case $N_h = 4$. Both distributions confirm that the damage position is almost always identified with high accuracy. How-

8. Chinesta *et al.*, *Virtual, Digital and Hybrid Twins: A New Paradigm in Data-Based Engineering and Engineered Data*. 2020

334. Hastie *et al.*, *The Elements of Statistical Learning*. 2009

335. Pedregosa *et al.*, *Scikit-learn: Machine Learning in Python*. 2011



ever, similarly to what observed for $\mathcal{L}_{\mathbf{h}}^{(1)}$, slightly less accurate identification occurs along the y-direction.

The statistics of the prediction errors for the considered choices of $N_h = \{2, 3, 4\}$ are presented in Tab. 5.2. The performance of the damage localization model is more than satisfactory even with $N_h = 2$. However, increasing N_h yields improvement in all the considered performance measures due to the increased representation capability of $\text{NN}_{\mathbf{h}}$. For any considered choice of N_h, the reported MAE values are smaller than, or equal to, half the size of the damaged region in both directions, with

Figure 5.5: L-shaped cantilever beam - Multidimensional scaling onto two dimensions of the $\text{NN}_{\mathbf{h}}$ embeddings using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$: (a) training data $\{\mathbf{h}_i = (h_1, h_2, h_3, h_4)_i^\top\}_{i=1}^{N_{\text{train}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (right) y-direction $\{y_{\Omega i}\}_{i=1}^{N_{\text{train}}}$; (b) testing data $\{\mathbf{h}_j = (h_1, h_2, h_3, h_4)_j^\top\}_{j=1}^{N_{\text{test}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (right) y-direction $\{y_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

$N_{\text{test}} = 800$	$N_h = 2$	$N_h = 3$	$N_h = 4$
N_k:	96	56	51
Weighting rule:	Uniform	Inverse	Inverse
$\text{MAE}(x_\Omega, \widehat{x}_\Omega)$:	0.120 m	0.091 m	0.082 m
$\text{MAE}(y_\Omega, \widehat{y}_\Omega)$:	0.152 m	0.116 m	0.105 m
$\text{Mean}(x_\Omega - \widehat{x}_\Omega)$:	0.037 m	0.038 m	0.028 m
$\text{Mean}(y_\Omega - \widehat{y}_\Omega)$:	0.073 m	0.078 m	0.047 m
$\text{StdDev}(x_\Omega - \widehat{x}_\Omega)$:	0.26 m	0.21 m	0.20 m
$\text{StdDev}(y_\Omega - \widehat{y}_\Omega)$:	0.30 m	0.27 m	0.25 m

Table 5.2: L-shaped cantilever beam - Accuracy of damage localization along the x and y directions using $\mathcal{L}_{\mathbf{h}}^{(2)}$ for $N_h = \{2, 3, 4\}$.

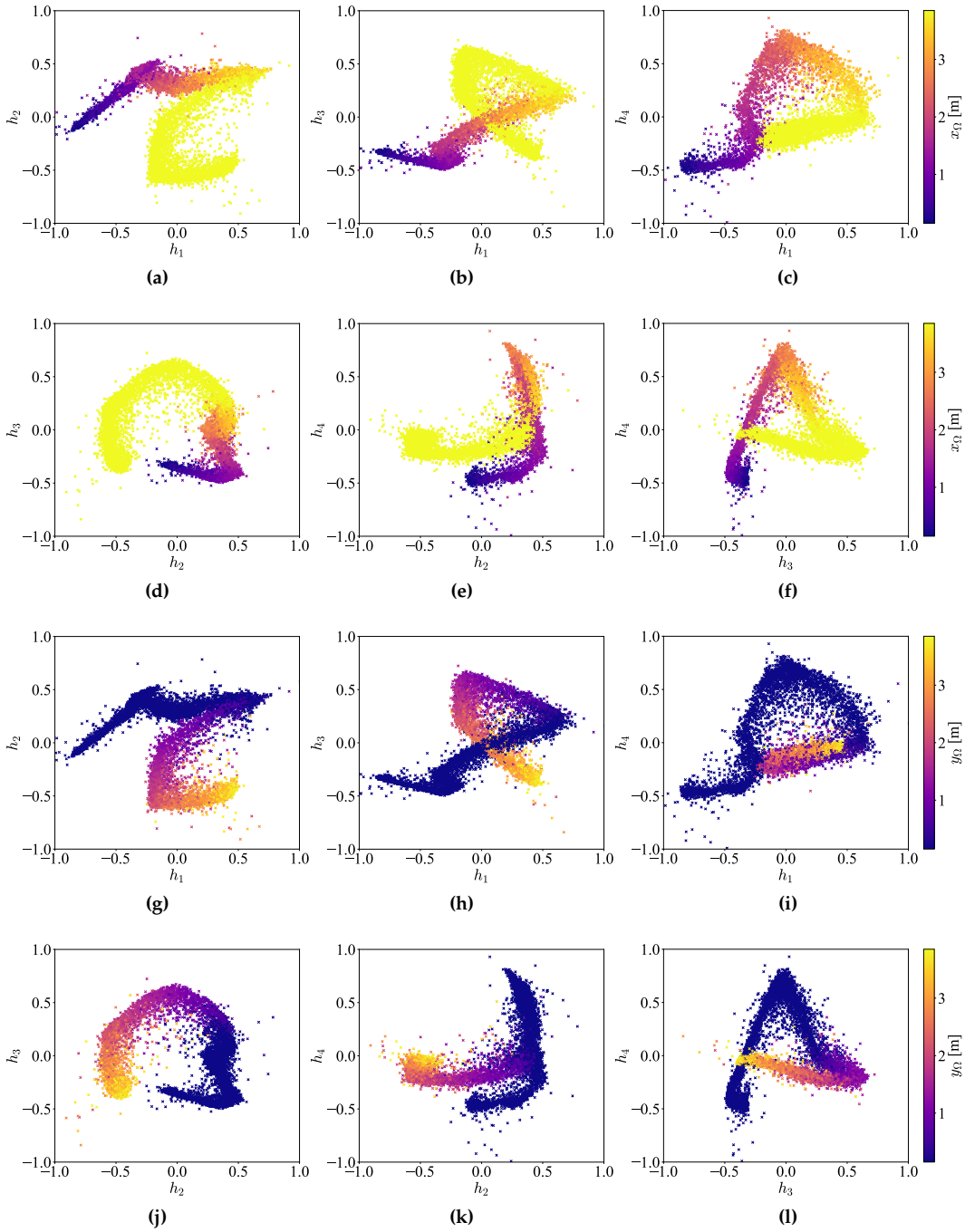


Figure 5.6: L-shaped cantilever beam - Low-dimensional embeddings of the training data $\{\mathbf{h}_i = (h_1, h_2, h_3, h_4)_i^\top\}_{i=1}^{N_{\text{train}}}$ using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$, plotted showing two components at a time against the correct damage position along: (a-f) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (g-l) y-direction $\{y_{\Omega i}\}_{i=1}^{N_{\text{train}}}$.

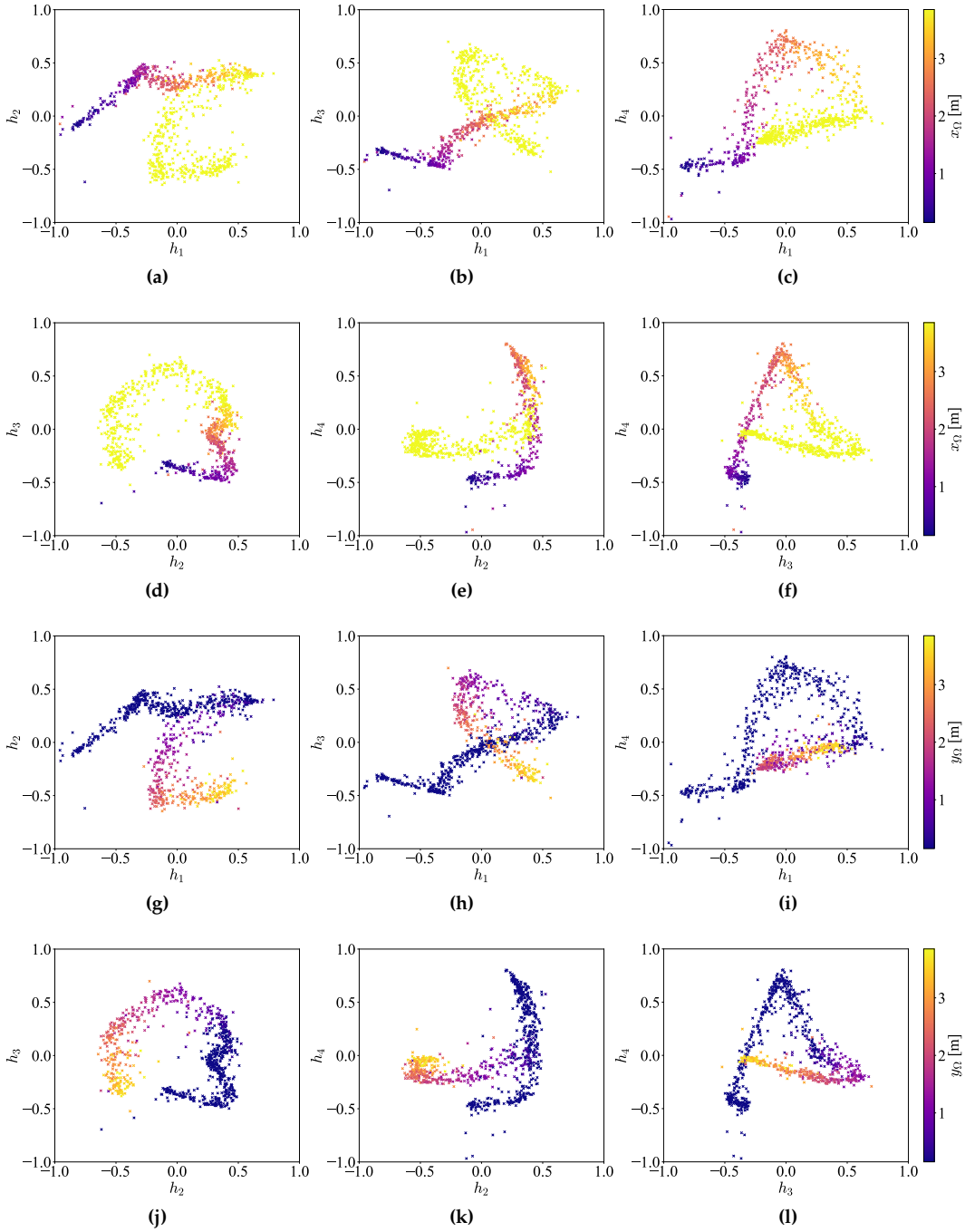


Figure 5.7: L-shaped cantilever beam - Low-dimensional embeddings of the testing data $\{\mathbf{h}_j = (h_1, h_2, h_3, h_4)_j^\top\}_{j=1}^{N_{\text{test}}}$ using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$, plotted showing two components at a time against the correct damage position along: (a-f) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (g-l) y-direction $\{y_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

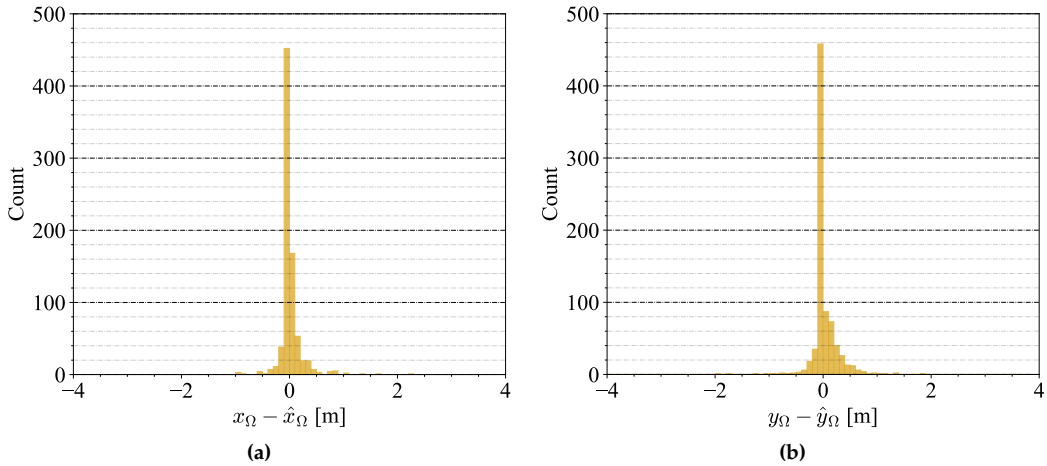


Figure 5.8: L-shaped cantilever beam - Distribution of prediction errors along the (a) x-direction and (b) y-direction, using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$.

the best accuracy level obtained selecting $N_h = 4$. The positive effect of increasing N_h is also reflected by the inverse distance weighting rule selected by the cross-validation method, which intuitively testifies the effectiveness of the learned mapping $\text{NN}_{\mathbf{h}}$. Specifically, employing $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$ and assuming that the prediction errors along the x and y directions follow Gaussian distributions, 95% of the testing population is expected to yield prediction errors bounded within $[-0.36, 0.42]$ m and $[-0.44, 0.54]$ m, respectively along the x and y directions. Despite the reported improvement in the damage localization capabilities for higher values of N_h, the computational time required to train and evaluate both $\text{NN}_{\mathbf{h}}$ and $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$ is not significantly affected. Indeed, a small change in the number of neurons in the output layer of $\text{NN}_{\mathbf{h}}$ has a negligible impact on the number of tunable parameters within $\Omega_{\mathbf{y}}$, while significantly enhancing its representation capability.

Loss function $\mathcal{L}_{\mathbf{h}}^{(3)}$. We now present the results obtained using the square-square loss function $\mathcal{L}_{\mathbf{h}}^{(3)}$, equipped with the metric built upon the cosine similarity defined in Eq. (5.6). In this case, the performance of $\text{NN}_{\mathbf{h}}$ is systematically analyzed for $N_h = \{3, 4, 5\}$. With respect to what reported for $\mathcal{L}_{\mathbf{h}}^{(2)}$, the size of the low-dimensional space is increased to account for the additional normalization constraint associated with the computation of the cosine similarity C_s. Anticipating the discussion of the results, and similarly to the findings for $\mathcal{L}_{\mathbf{h}}^{(2)}$, the best results are provided with $N_h = 5$.

The 2D MDS representations of the low-dimensional embeddings provided by $\text{NN}_{\mathbf{h}}$ for the training and testing data in the case of $N_h = 5$ are presented in Fig. 5.9. Similar to the MDS

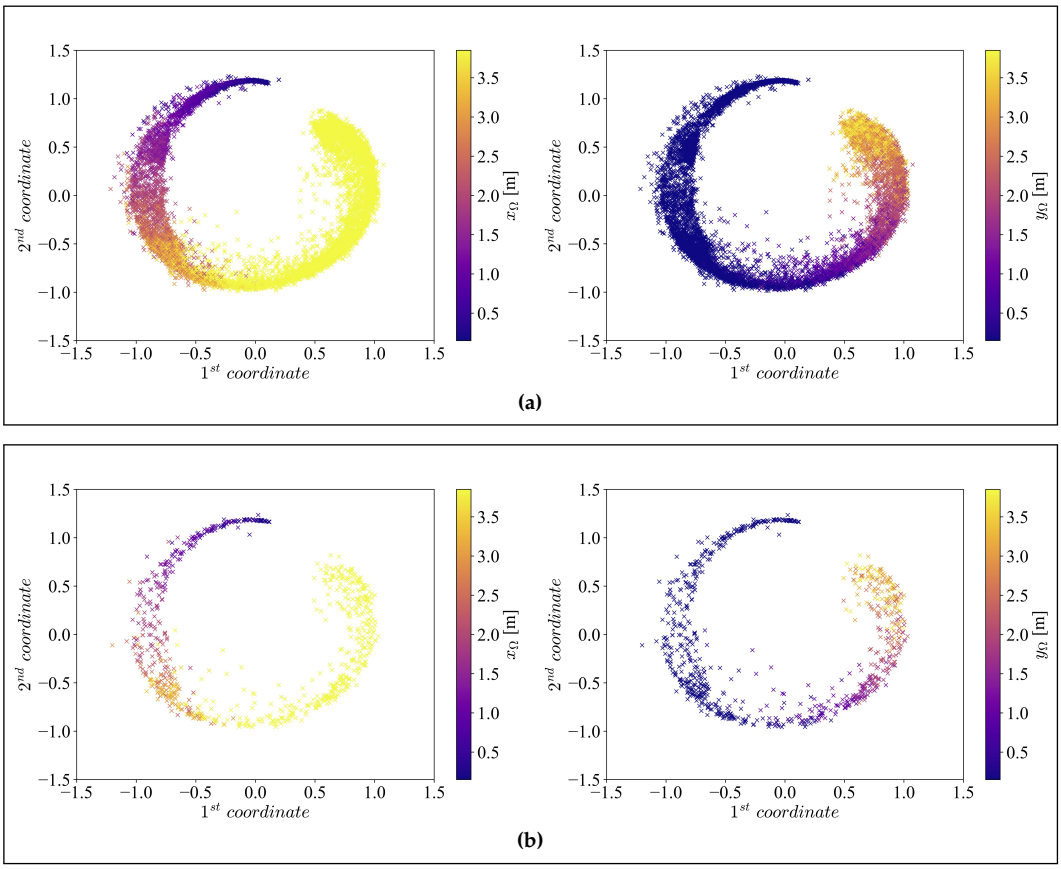
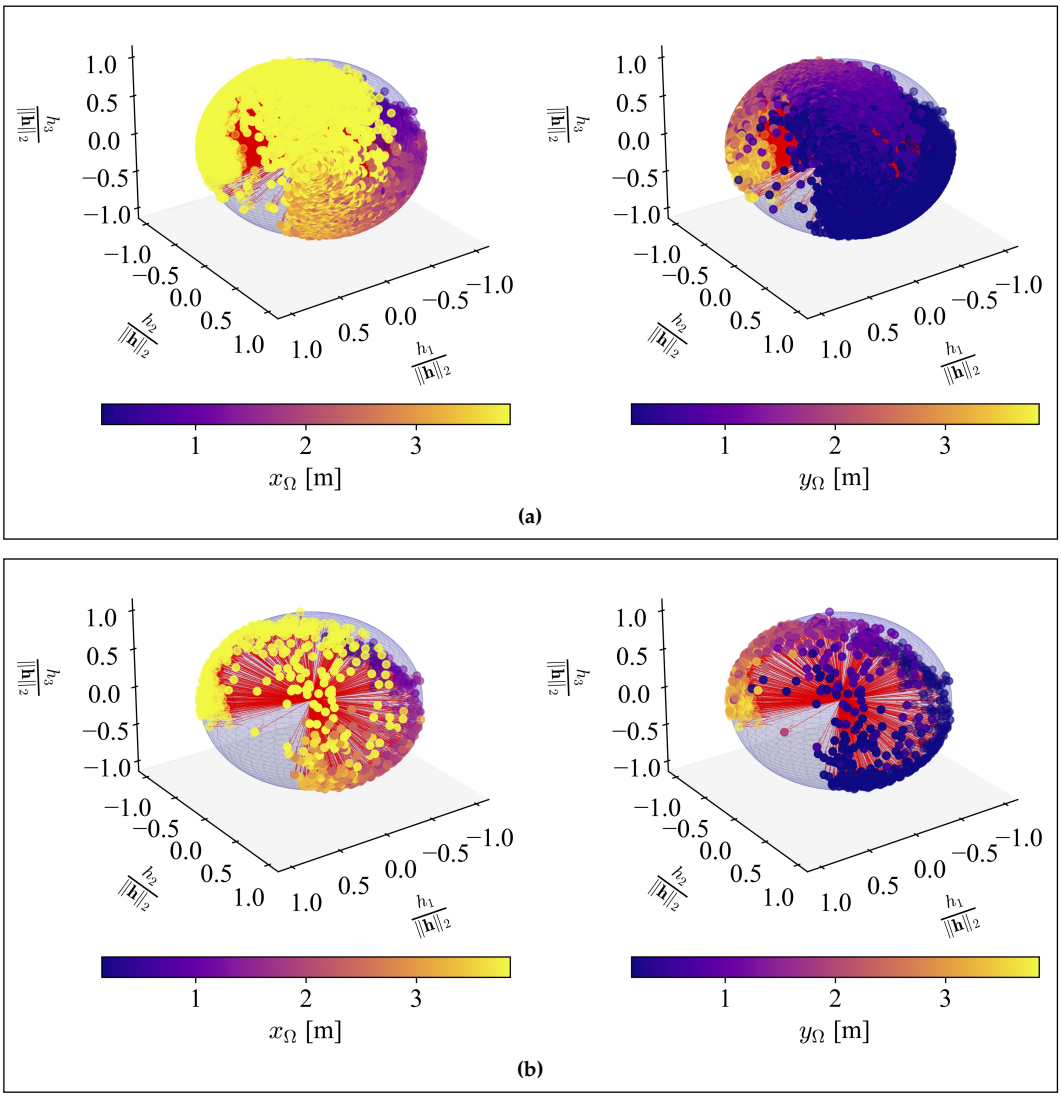


Figure 5.9: L-shaped cantilever beam - Multidimensional scaling onto two dimensions of the $\text{NN}_\mathbf{h}$ embeddings using $\mathcal{L}_\mathbf{h}^{(3)}$ with $N_h = 5$: (a) training data $\{\mathbf{h}_i = (h_1, h_2, h_3, h_4, h_5)_i^\top\}_{i=1}^{N_{\text{train}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (right) y-direction $\{y_{\Omega i}\}_{i=1}^{N_{\text{train}}}$; (b) testing data $\{\mathbf{h}_j = (h_1, h_2, h_3, h_4, h_5)_j^\top\}_{j=1}^{N_{\text{test}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (right) y-direction $\{y_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

representations obtained for $\mathcal{L}_\mathbf{h}^{(2)}$, the overall shape defined by the scatter plots does not resemble the structural layout. However, the resulting manifold effectively encodes the sensitivity of the processed measurements to the position of damage. It is worth noting how this manifold increases in size as the damage position moves away from the clamped side, testifying the associated smaller impact on the sensor recordings.

For completeness, in Fig. 5.10 we report additional scatter plots showing the source low-dimensional embeddings $\{\mathbf{h}_i = (h_1, h_2, h_3)_i^\top\}_{i=1}^{N_{\text{train}}}$ and $\{\mathbf{h}_j = (h_1, h_2, h_3)_j^\top\}_{j=1}^{N_{\text{test}}}$, as obtained with $N_h = 3$ for the training and testing data, respectively. Only the case associated with $N_h = 3$ is employed to illustrate the source low-dimensional embeddings, as it allows for the complete representation of an exemplary metric space built upon the cosine similarity on a unit sphere. As for the MDS representations, it is interesting to note how embeddings associated with different damage conditions feature a relative rotation in the low-dimensional space.

When training $\text{NN}_\mathbf{h}$ on $\mathcal{L}_\mathbf{h}^{(3)}$, the same metric built upon the cosine similarity C_s must be consistently employed for the



computation of distances during the training and evaluation of $\mathcal{M}^{\mathbf{y}}_{\text{k-NN}}$. The distributions of the prediction errors along the x and y directions for the testing data are presented in Fig. 5.11, focusing on the case $N_h = 5$. The observed values of prediction errors are distributed similarly to what reported for $\mathcal{L}^{(1)}_{\mathbf{h}}$ and $\mathcal{L}^{(2)}_{\mathbf{h}}$, both in the x and y directions. However, it can be qualitatively appreciated that damage localizations are slightly less accurate for the present case of $\mathcal{L}^{(3)}_{\mathbf{h}}$.

The statistics of the prediction errors for the considered choices of $N_h = \{3, 4, 5\}$ are reported in Tab. 5.3. In contrast to what observed for $\mathcal{L}^{(2)}_{\mathbf{h}}$, in this case increasing N_h only yields a partial improvement in the damage localization capabilities enabled by $\text{NN}_{\mathbf{h}}$. The performance indicators along the y-

Figure 5.10: L-shaped cantilever beam - Normalized $\text{NN}_{\mathbf{h}}$ embeddings using $\mathcal{L}^{(3)}_{\mathbf{h}}$ with $N_h = 3$: (a) training data $\{\mathbf{h}_i = (h_1, h_2, h_3)^\top_i\}^{N_{\text{train}}}_{i=1}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}^{N_{\text{train}}}_{i=1}$ and (right) y-direction $\{y_{\Omega i}\}^{N_{\text{train}}}_{i=1}$; (b) testing data $\{\mathbf{h}_j = (h_1, h_2, h_3)^\top_j\}^{N_{\text{test}}}_{j=1}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}^{N_{\text{test}}}_{j=1}$ and (right) y-direction $\{y_{\Omega j}\}^{N_{\text{test}}}_{j=1}$.

	$N_h = 3$	$N_h = 4$	$N_h = 5$
N_k:	120	103	82
Weighting rule:	Uniform	Uniform	Inverse
MAE($x_\Omega, \widehat{x}_\Omega$):	0.143 m	0.132 m	0.125 m
MAE($y_\Omega, \widehat{y}_\Omega$):	0.148 m	0.146 m	0.148 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.032 m	0.038 m	0.032 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.074 m	0.075 m	0.061 m
StdDev($x_\Omega - \widehat{x}_\Omega$):	0.27 m	0.26 m	0.26 m
StdDev($y_\Omega - \widehat{y}_\Omega$):	0.27 m	0.27 m	0.27 m

Table 5.3: L-shaped cantilever beam - Accuracy of damage localization along the x and y directions using $\mathcal{L}_{\mathbf{h}}^{(3)}$ for $N_h = \{3, 4, 5\}$.

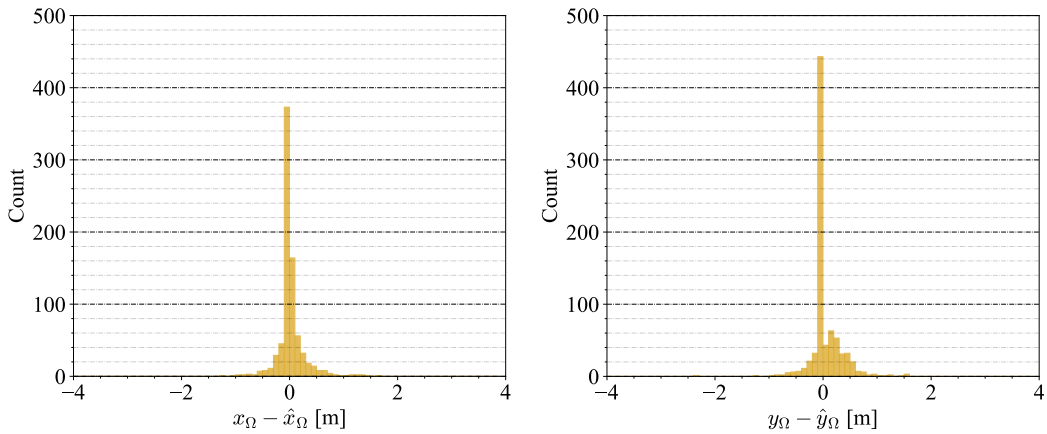


Figure 5.11: L-shaped cantilever beam - Distribution of prediction errors along the (a) x-direction and (b) y-direction, using $\mathcal{L}_{\mathbf{h}}^{(3)}$ with $N_h = 5$.

direction do not show improvement while increasing N_h, and those along the x-direction improve only to a limited extent compared to what observed for $\mathcal{L}_{\mathbf{h}}^{(2)}$. As anticipated, the best accuracy level is reported for the largest size $N_h = 5$ of the low-dimensional space. By using this setup to train $\text{NN}_{\mathbf{h}}$, 95% of the testing population is expected to yield prediction errors bounded within $[-0.48, 0.54]$ m and $[-0.47, 0.59]$ m, respectively along the x and y directions.

Discussion of the results. The results obtained for the present case study are more than satisfactory across all the considered alternatives of loss function. Regardless of the chosen training configuration, $\text{NN}_{\mathbf{h}}$ consistently performs better along the x-direction, due to the smaller sensitivity of sensor recordings to damage when damage is located near the free-end of the beam. Moreover, for all considered sizes of the low-dimensional space, the three loss functions yield MAE values for prediction smaller than, or equal to, half the size of the damaged region in both directions, which is a remarkable result.

The least accurate damage localization outcomes are obtained when training $\text{NN}_{\mathbf{h}}$ by minimizing $\mathcal{L}_{\mathbf{h}}^{(3)}$, with the associated performance being almost insensitive to the selected size N_h of

N_k:	51	Weighting rule:	Inverse
MAE($x_\Omega, \widehat{x}_\Omega$):	0.122 m	MAE($y_\Omega, \widehat{y}_\Omega$):	0.133 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.045 m	Mean($y_\Omega - \widehat{y}_\Omega$):	0.048 m
StdDev($x_\Omega - \widehat{x}_\Omega$):	0.30 m	StdDev($y_\Omega - \widehat{y}_\Omega$):	0.31 m

Table 5.4: L-shaped cantilever beam - Accuracy of damage localization for damage regions of varying size using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$.

the low-dimensional space. Despite the expectation that using $\mathcal{L}_{\mathbf{h}}^{(3)}$ would improve the representation capability of $\text{NN}_{\mathbf{h}}$, we argue that the metric built upon C_s excessively constraints the learning process. On the other side, the most accurate results are obtained when training $\text{NN}_{\mathbf{h}}$ on $\mathcal{L}_{\mathbf{h}}^{(2)}$. In this case, increasing N_h provides a non-negligible improvement of the localization outcomes. The positive effect of increasing N_h is also reflected in the inverse distance weighting rule selected by the cross-validation method, which intuitively testifies the effectiveness of the learned mapping. Finally, training $\text{NN}_{\mathbf{h}}$ on $\mathcal{L}_{\mathbf{h}}^{(1)}$ provides slightly worse, yet remarkable, localization outcomes compared to $\mathcal{L}_{\mathbf{h}}^{(2)}$. However, the low-dimensional manifold enabled by the learned mapping is easier to interpret and visualize.

Robustness Assessment. To assess the robustness of the proposed methodology against uncertainties related to the size of the damaged region, we test the best-performing configuration using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$ on an additional test set comprising $N_{\text{test}} = 800$ pseudo-experimental instances. Similar to the previous cases, these instances are derived from FOM solutions corrupted with additive Gaussian noise to achieve an SNR of 80. However, in this case, we vary not only the magnitude of the stiffness reduction $\delta \in [10\%, 25\%]$ but also the *longitudinal thickness* of the subdomain $\Omega_{\mathbf{y}}$ wherein the stiffness is reduced. Instead of keeping the longitudinal size of $\Omega_{\mathbf{y}}$ fixed at 0.3 m, we treat it as an additional parameter included in $\boldsymbol{\mu}$, and taking values in $[0.2, 0.45]$ m. It is important to note that this variation applies only to the testing data, while the standard value of 0.3 m is used for training, as in the previous cases.

The obtained results are summarized in Tab. 5.4, presenting the statistics of the associated prediction errors. These statistics closely align to those reported in Tab. 5.2. In this case as well, the MAE values for prediction are smaller than half the size of the damaged region, with the associated mean values close to zero in both directions. Despite slightly higher deviations compared to the reference testing condition, 95% of the testing population is expected to yield prediction errors bounded within $[-0.54, 0.63]$ m and $[-0.56, 0.66]$ m, respectively along the x and y directions. This is a remarkable result, especially considering damaged regions varying in size and damage level.

5.4 DAMAGE LOCALIZATION: RAILWAY BRIDGE

This second case study aims to assess the proposed methodology in a pseudo-real setting, involving the railway bridge in Fig. 5.12. This integral concrete bridge is situated along the Bothnia line in the coastal area of Hörnefors (Sweden). It spans 15.7 m, with a free height of 4.7 m and a width of 5.9 m (edge beams excluded). The wing walls extend in the longitudinal direction, reaching a length of 8 m. The structural elements have a thicknesses of 0.5 m for the deck, 0.7 m for the frame walls, and 0.8 m for the wing walls. The bridge is founded on two plates connected by two stay beams and supported by pile groups. The concrete is of class C35/45, with mechanical properties: $E = 34$ GPa, $\nu = 0.2$, $\rho = 2500$ kg/m^3. The superstructure consists of a single track with sleepers spaced 0.65 m apart, resting on a 0.6 m deep, 4.3 m wide ballast layer with a density of $\rho_B = 1800$ kg/m^3. The geometrical and mechanical data are adapted from previous a research involving the relevant soil-structure interaction [336,337]. Refer to Fig. 5.13 for further details.

The bridge is subjected to the transit of *Gröna Tåget* trains, as depicted in Fig. 5.14, traveling at a speed of $v \in [160, 215]$ km/h. We specifically consider trains composed of two wagons, totaling 8 axles. Each axle carries a mass of $m_A \in [16, 22]$ ton.

336. Arvidsson *et al.*, *Dynamic analysis of a portal frame railway bridge using frequency dependent soil structure interaction.* 2011

337. Ülker-Kaustell, *Some aspects of the dynamic soil-structure interaction of a portal frame railway bridge.* 2009

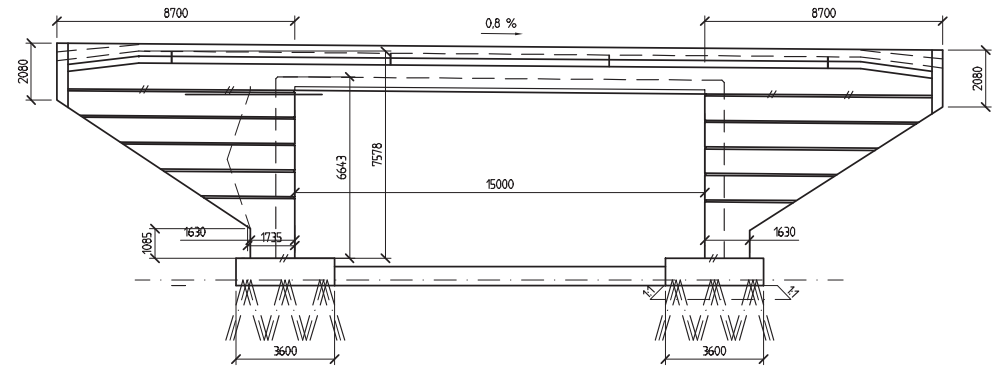
*Our goal is to identify damage patterns that result in a loss of stiffness within localized areas of the structure. In general, these damages can arise from long-term material degradation caused by chemical, physical, or mechanical processes, as well as sudden changes due to discrete damage events [338]. The integral design of the Hörnefors bridge minimizes the ingress of moisture, debris, and corrosive agents into the structure, thus enhancing its long-term durability. The types of damage affecting these bridges vary depending on structural layout, materials, construction practices, environmental conditions, and traffic loads. Some damage patterns that we may seek to describe through a localized stiffness reduction include: concrete cracking due to thermal gradients, freeze-thaw cycles, and overloading; gradual concrete degradation due to the chemical reaction between reactive silica aggregates and alkalis in the cement paste, potentially leading to cracking and spalling; concrete cracking and structural damage due to stress concentrations originating from differential foundational settlements; and erosion of concrete surfaces due to vehicular collisions or exposure to environmental factors. We recognize that not all of these damages may significantly affect the effective stiffness, and we can not claim to identify damages not observable through vibration recordings of the structural response. Nonetheless, assessing the methodology in such a scenario is still deemed of interest.

5.4.1 DATASET ASSEMBLY AND TRAINING OPTIONS

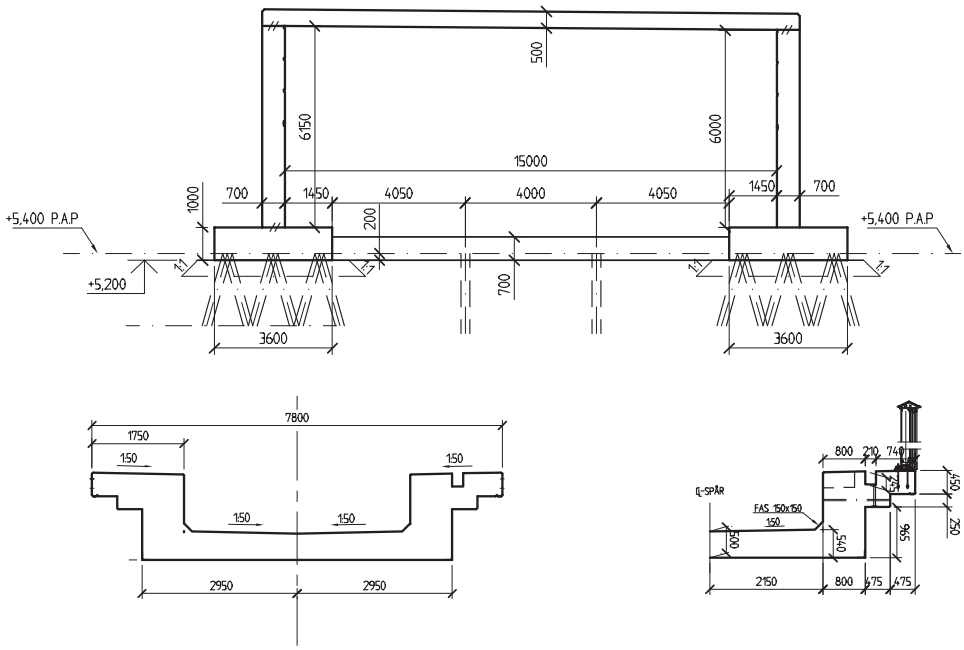
The monitoring system is deployed to provide displacement recordings $\mathbf{U}(\boldsymbol{\mu}) = [\mathbf{u}_1(\boldsymbol{\mu}), \ldots, \mathbf{u}_{N_u}(\boldsymbol{\mu})]$, corresponding to the $N_u = 10$ DOFs highlighted in Fig. 5.15. Each recording spans a time interval $(0, T = 1.5$ s$)$, with a sampling frequency of $f_s = 400$ Hz. This setup ensures the recording of train passages even at the lowest speed of 160 km/h, while adequately capturing the structural response at the maximum speed of 215 km/h.

Given the adopted methodological perspective, we recognize that achieving an experimental validation on such a structural system would probably require additional modeling efforts compared to what is described in the following*. In this case, damage is simulated through localized stiffness reductions that

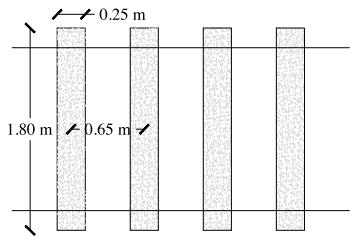
Figure 5.12: Hörnefors railway bridge.



(a)



(b)



(c)

Figure 5.13: Railway bridge - Geometrical data: (a) longitudinal view (source: ELU Konsult); (b) longitudinal and transversal cross-sections (source: ELU Konsult); (c) sleepers layout.

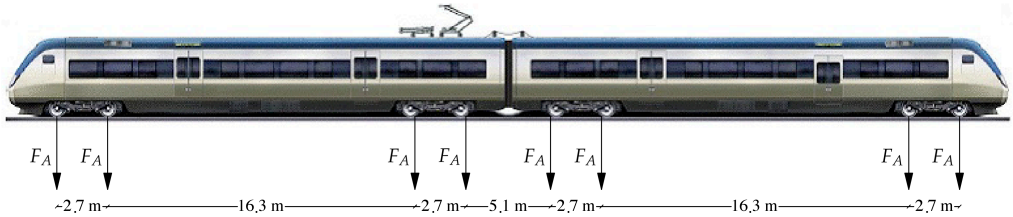


Figure 5.14: Gröna Tåget train type.

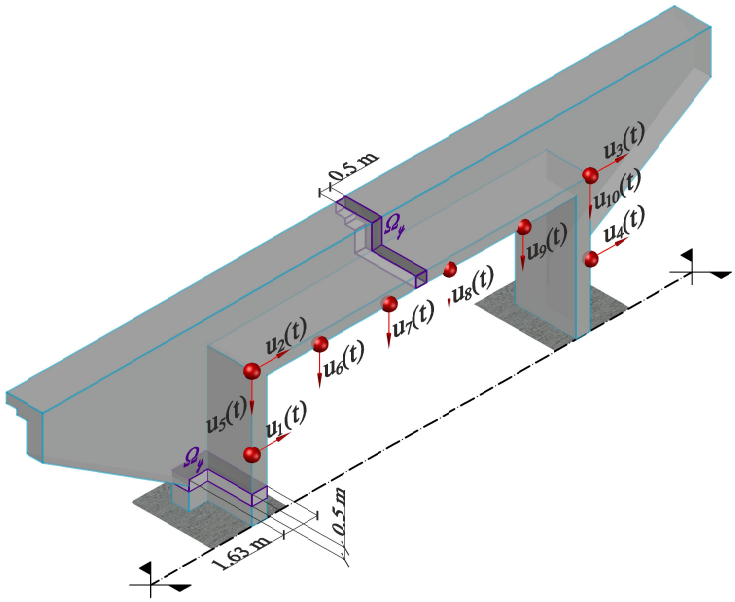


Figure 5.15: Railway bridge - Details of synthetic recordings related to displacements $u_1(t), \ldots, u_{10}(t)$, and damageable region $\Omega_{\mathbf{y}}$.

can occur anywhere over the two lateral frame walls and the deck, within specific subdomains $\Omega_{\mathbf{y}}$. The layout of these subdomains differs between the two cases, as shown Fig. 5.15. The target position of $\Omega_{\mathbf{y}}$ is parametrized through $\mathbf{y} = (x_\Omega, z_\Omega)^\top$, with either x_Ω or z_Ω varying in the ranges of $[-0.115, 16.63]$ m and $[0.25, 6.25]$ m, respectively. The stiffness reduction within $\Omega_{\mathbf{y}}$ can occur with a magnitude $\delta \in [30\%, 40\%]$, and is kept fixed while a train travels across the bridge.

The FOM is established through a FE discretization using linear tetrahedral elements, as shown in Fig. 5.16. To enable a smooth transmission of the traveling load, a reduced element size of 0.15 m is employed for the deck, while an element size of 0.80 m is used elsewhere, resulting in a total of $N_{\text{FE}} = 17,292$ DOFs. The ballast layer is accounted for through an increased density for the deck and the edge beams. The embankments are modeled with distributed springs over the surfaces facing the ground, implemented as a *Robin* mixed boundary condition (elastic coefficient $k_{\text{robin}} = 10^8$ N/m^3). Damping is disregarded.

The load from each axle is transmitted to the structure through 25 sleepers and propagated through the ballast with a slope 4 : 1, according to Eurocode 1 [339]. Each sleeper provides

339. European Committee for Standardization, *Part 2: Traffic loads on bridges*. 2003

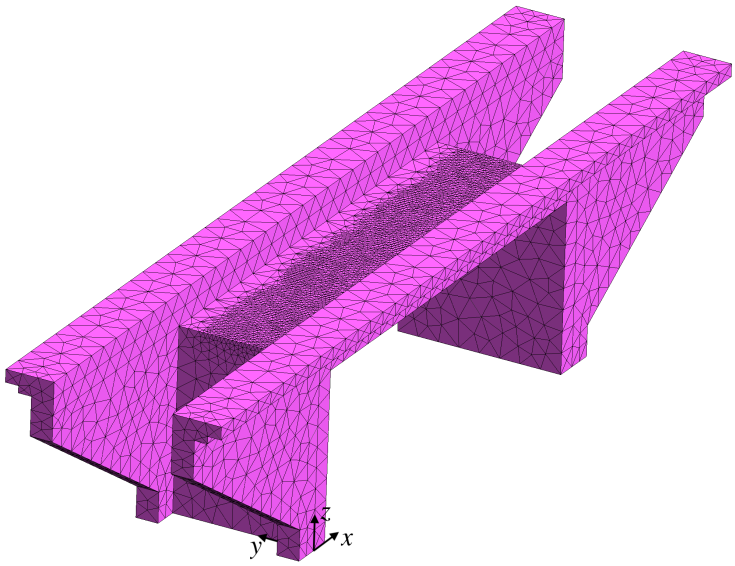


Figure 5.16: Railway bridge - Finite element discretization.

a load to a surface of $(0.55\times2.1)\,\mathrm{m}^2$. Denoting the force associated with the axle-carried mass m_A as F_A, the maximum pressure that an axle can exert on the structure is $p_{\max} = \frac{F_A}{(0.55\times2.1)\ \mathrm{m}^2}$. The entire moving load system, including contributions from all 8 axles traveling over the 25 sleepers, is modeled as:

$$P(x,t,v) = \sum_{n_S=1}^{25}\sum_{n_A=1}^{8} \mathcal{A}^{n_S}(x)\mathcal{A}^{n_S}_{n_A}(t,v)p_{\max}. \tag{5.10}$$

Herein, $\mathcal{A}^{n_S}(x)$ is the *space activation function* for the n_S-th sleeper, and $\mathcal{A}^{n_S}_{n_A}(t,v)$ is the *time-modulating function* for the n_A-th axle, with $n_S = 1,\dots,25$ and $n_A = 1,\dots,8$. In particular, the space activation function $\mathcal{A}^{n_S}(x)$ accounts for the load applied on the projection of the n_S-th sleeper and, for $n_S = 1,\dots,25$, reads:

$$\mathcal{A}^{n_S}(x) = \mathscr{H}\Big(x - (x_{n_S} - \frac{0.55\ \mathrm{m}}{2})\Big) - \mathscr{H}\Big(x - (x_{n_S} + \frac{0.55\ \mathrm{m}}{2})\Big), \tag{5.11}$$

where x_{n_S} is the abscissa of the axis of the n_S-th sleeper, and $\mathscr{H} : \mathbb{R} \mapsto \mathbb{R}$ is the *Heavyside function* defined as:

$$\mathscr{H}(a) = \begin{cases} 1 & \text{if } a \geq 0, \\ 0 & \text{if } a < 0. \end{cases} \tag{5.12}$$

On the other hand, the time-modulating function $\mathcal{A}^{n_S}_{n_A}(t,v)$ enables modulating the pressure value as a function of time and axle speed, for $n_A = 1,\dots,8$ and $n_S = 1,\dots,25$, according to:

$$\begin{aligned}\mathcal{A}^{n_S}_{n_A}(t,v) = \Big[\mathscr{H}\Big(t - \frac{x_{n_S-1} + x^0_{n_A}}{v}\Big) - \\ \mathscr{H}\Big(t - \frac{x_{n_S+1} + x^0_{n_A}}{v}\Big)\Big]\Big(1 - \frac{|t - \frac{x_{n_S}+x^0_{n_A}}{v}|}{\frac{0.65\ \mathrm{m}}{v}}\Big),\end{aligned} \tag{5.13}$$

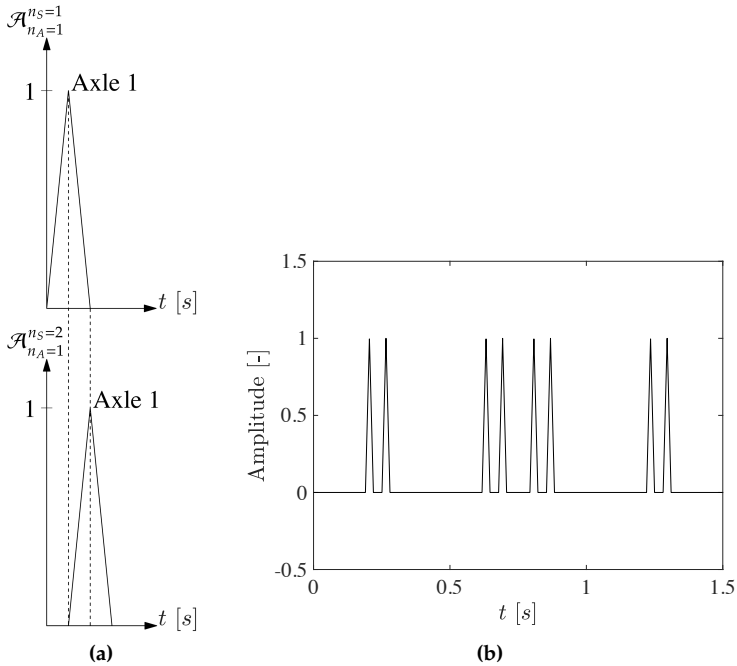


Figure 5.17: Railway bridge - Time-modulating function: (a) an example related to the effect of the 1st axle on the 1st and 2nd sleepers; (b) behavior during the time interval $(0, T)$ for the passage of 8 axles over the midspan sleeper at a train speed of v = 160 km/h.

where $x^0_{n_A}$ is the abscissa of the n_A-th axle at time $t = 0$. The first term allows for the activation of the load transmission when the n_A-th axle is between the axes of the $(n_S - 1)$-th and $(n_S + 1)$-th sleepers. The second term, instead, allows for a piece-wise linear variation between 0 and $p_{\max}$ and between $p_{\max}$ and 0, respectively, when the n_A-th axle is traveling from x_{n_S-1} to x_{n_S} and from x_{n_S} to x_{n_S+1}. According to Eq. (5.13), the maximum value of pressure on a sleeper projection is attained when an axle is crossing its axis, and it becomes null when the axle is crossing the axis of the previous or next sleeper, with a linear variation in between. Fig. 5.17a shows an example for the first axle when it is traveling from the 1st to the 2nd sleeper. Fig. 5.17b instead shows the behavior of the time-modulating function for the midspan sleeper, subjected to the passage of a train at a speed of v = 160 km/h. This graph is useful to display an exemplary history of the external loading over the time interval $(0, T)$.

In this case, the basis matrix $\mathbf{W}$ underlying the ROM is obtained from a snapshot matrix $\mathbf{S}$ assembled through $N_P = 300$ evaluations of the FOM over $(0, T)$, for varying configurations of the input parameters $\boldsymbol{\mu} = (v, m_A, \mathbf{y}, \delta)^\top$. By setting the error tolerance to $\epsilon = 10^{-3}$, $N_{\mathrm{RB}} = 528$ POMs are considered in place of the original 17,292 DOFs. The CPU time required for each simulation accordingly decreases from 82.95 s to 12.50 s, yielding a speed-up of about 7 times. It is worth noting that the number of POMs is greater than that reported for the L-shaped

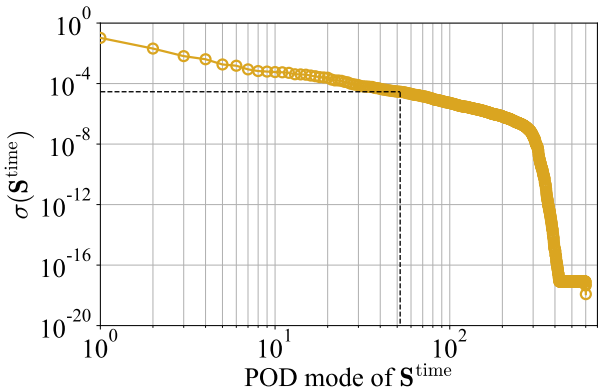


Figure 5.18: Railway bridge - Exemplary spectrum of singular values for a proper orthogonal decomposition on solution snapshots collected by evaluating the full-order model over the time interval $(0, T)$ (axes in logarithmic scale).

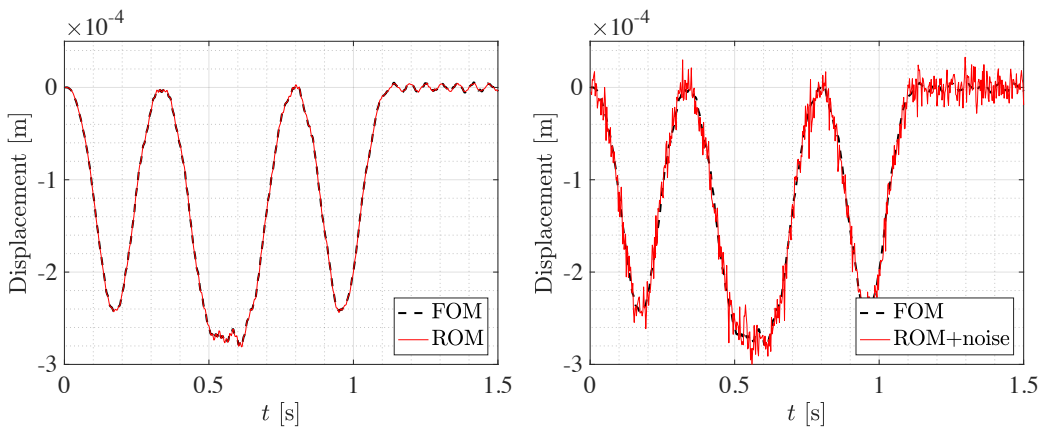


Figure 5.19: Railway bridge - Example of vertical displacement time histories at midspan: comparison between (a) full-order solution and reduced-order approximation, and (b) full-order solution and noise-corrupted reduced-order approximation.

cantilever beam, reflecting the need for the ROM to fully capture the specific response of the structure under the effect of the traveling load for various damage scenarios. To provide insights into this aspect, Fig. 5.18 presents an exemplary *spectrum of the singular values* obtained by performing a POD over a sample FOM solution for the time interval $(0, T)$. We highlight that the graph shows a satisfactory decay only in its final part, indicating the need for a large number of POMs to meet the energy content criterion.

The dataset $\mathcal{D}_{\text{LOC}}$ is populated with $N_{\text{train}} = 8000$ instances collected through the ROM. To account for sensor noise, all the data are contaminated with independent, identically distributed Gaussian noise, resulting in an SNR equal to 120. These data instances are then augmented to assemble $\mathcal{D}^{\text{P}}_{\text{LOC}}$ with $N_{+} = N_{-} = 10$ pairs, after setting $\overline{\mathcal{E}_{\mathbf{y}}} = 0.25$ m. The testing set consists of $N_{\text{test}} = 800$ pseudo-experimental instances derived from noise-corrupted FOM solutions. Time histories of the vertical displacement at midspan are reported in Fig. 5.19 to compare the FOM and ROM solutions in the absence or presence of the

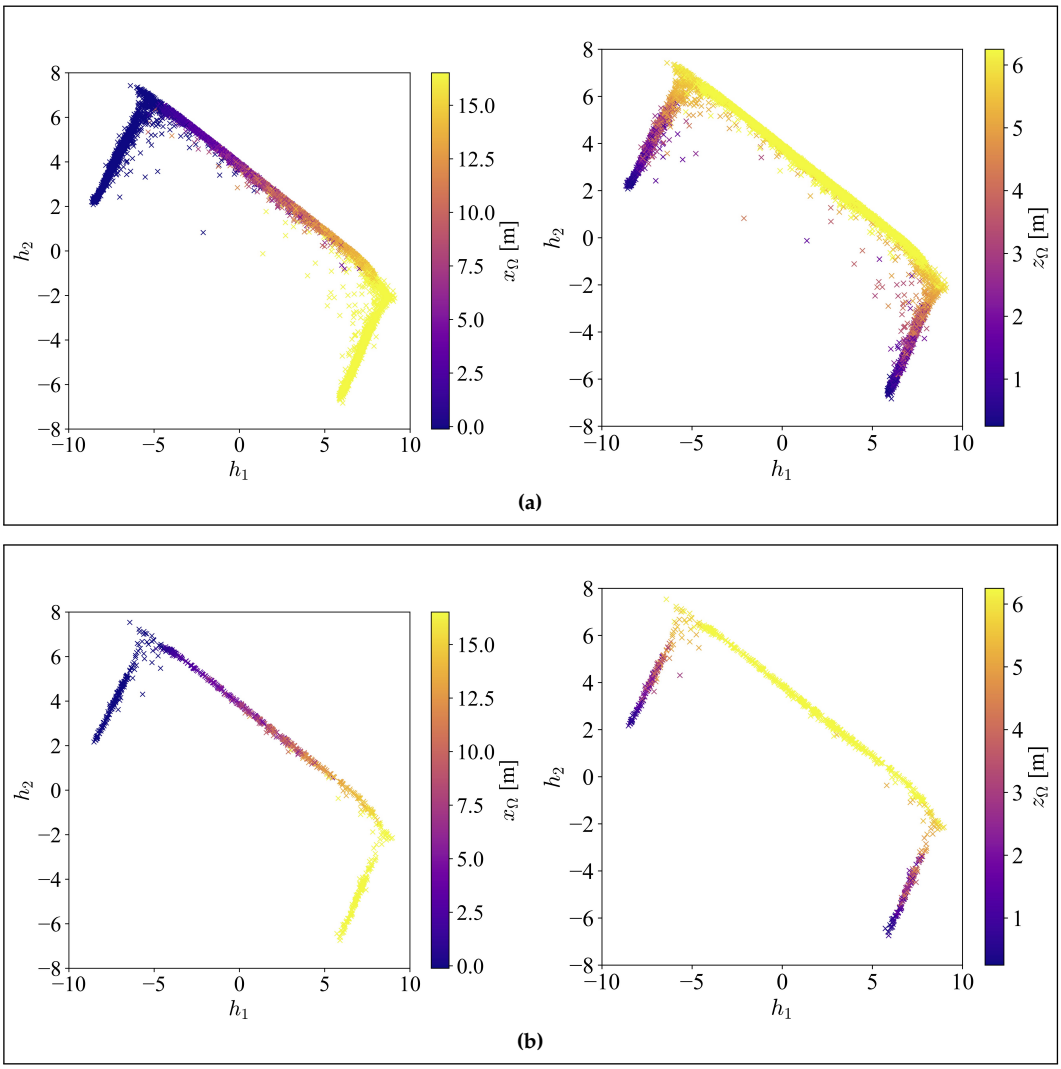


Figure 5.20: Railway bridge - Low-dimensional embeddings using $\mathcal{L}_{\mathbf{h}}^{(1)}$: (a) training data representations $\{\mathbf{h}_i = (h_1, h_2)_i^\top\}_{i=1}^{N_{\text{train}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (right) z-direction $\{z_{\Omega i}\}_{i=1}^{N_{\text{train}}}$; (b) testing data representations $\{\mathbf{h}_j = (h_1, h_2)_j^\top\}_{j=1}^{N_{\text{test}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (right) z-direction $\{z_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

added measurement noise, to assess both the ROM accuracy and the potential impact of noise on the handled structural response to the loading.

5.4.2 Localization outcome: numerical results

In light of the findings from the L-shaped cantilever beam case, only the loss functions $\mathcal{L}_{\mathbf{h}}^{(1)}$ and $\mathcal{L}_{\mathbf{h}}^{(2)}$ are considered for the present investigation.

Loss function $\mathcal{L}_{\mathbf{h}}^{(1)}$. We first present the results obtained by training $\text{NN}_{\mathbf{h}}$ using the MSE-based loss function $\mathcal{L}_{\mathbf{h}}^{(1)}$. Similar to the L-shaped beam case, the size of the low-dimensional space is set to $N_h = 2$. The low-dimensional embeddings for

N_k:	42	Weighting rule:	Uniform
MAE($x_\Omega, \widehat{x}_\Omega$):	0.692 m	MAE($z_\Omega, \widehat{z}_\Omega$):	0.306 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.043 m	Mean($z_\Omega - \widehat{z}_\Omega$):	0.019 m
StdDev($x_\Omega - \widehat{x}_\Omega$):	1.32 m	StdDev($z_\Omega - \widehat{z}_\Omega$):	0.60 m

Table 5.5: Railway bridge - Accuracy of damage localization along the x and z directions using $\mathcal{L}_{\mathbf{h}}^{(1)}$.

the training and testing data are presented in Fig. 5.20a and Fig. 5.20b, respectively. These graphs demonstrate that the mapping is properly learned, as the overall shape defined by the scatter plots clearly reflects the geometrical parametrization of the damage position along the x and z directions. In this case as well, it can be claimed that $\text{NN}_{\mathbf{h}}$ does not incur overfitting issues, as the low-dimensional representations obtained with the training and testing data show an extremely good agreement.

Regarding damage localization, $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$ is devised to provide predictions for unseen data based on $N_k = 42$ nearest neighbors and a uniform weighting rule, as selected by a 10-fold cross-validation. The distributions of the prediction errors along the x and z directions are presented in Fig. 5.21. The two distributions offer a first qualitative indication that the damage position is almost always identified with great accuracy, as the observed values are concentrated in a relatively narrow range around the origin. The associated statistics are presented in Tab. 5.5, showing significantly larger inaccuracies in locating damage along the x-direction: both the MAE and the standard deviation values are more than doubled compared to the z-direction. Despite that, the damage position is often provided with high accuracy. Assuming that the prediction errors along the x and z directions follow Gaussian distributions, it is expected that 95% of the testing population will yield prediction errors lying within the intervals $[-2.54, 2.63]$ m and $[-1.16, 1.19]$ m, respectively along the x and z directions.

Loss function $\mathcal{L}_{\mathbf{h}}^{(2)}$. We now present the results obtained training $\text{NN}_{\mathbf{h}}$ on the Square-Square loss function $\mathcal{L}_{\mathbf{h}}^{(2)}$ equipped with the Euclidean metric. Based on the findings presented in Sec. 5.3.2, the size of the low-dimensional space is directly set to $N_h = 4$, yielding the best-performing configuration for $\mathcal{L}_{\mathbf{h}}^{(2)}$. As for to the L-shaped cantilever beam case, $N_h = 4$ equals the number of independent parameters responsible for the data variability — also in this case, the position of the damaged subdomain $\Omega_{\mathbf{y}}$ can be represented as an abscissa running along the axis of the structure.

The 2D MDS representations of the low-dimensional embeddings provided by $\text{NN}_{\mathbf{h}}$ for the training and testing data are reported in Fig. 5.22a and Fig. 5.22b, respectively. Consistently

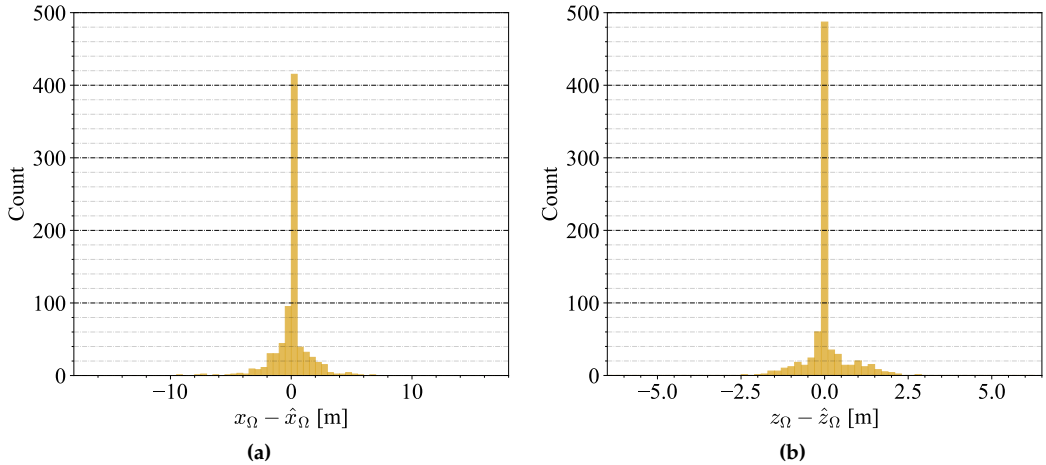


Figure 5.21: Railway bridge - Distribution of prediction errors along the (a) x-direction and (b) z-direction, using $\mathcal{L}_{\mathbf{h}}^{(1)}$.

with what observed for $\mathcal{L}_{\mathbf{h}}^{(1)}$, these plots qualitatively highlight that the damage position is encoded in the low-dimensional space with higher accuracy along the z-axis, while the portion of the manifold describing the variation along the x-axis appears more confused and increased in size.

Regarding $\mathcal{M}_{\text{k-NN}}^{\mathbf{y}}$, the cross-validation strategy selects $N_k = 51$ as the number of nearest neighbors and an inverse distance weighting rule. The damage localization outcomes are presented in Fig. 5.23, illustrating the distributions of prediction errors along the x and z directions. Qualitatively, the two distributions feature a smaller variance compared to those obtained leveraging $\mathcal{L}_{\mathbf{h}}^{(1)}$. This observation is supported by the associated statistics presented in Tab. 5.6, reporting improvements across all the considered performance measures, especially along the x-direction: the MAE values improve from 0.692 m to 0.453 m and from 0.306 m to 0.260 m, respectively along the x and z directions. According to these statistics, it is expected that 95% of the testing population will yield prediction errors within the intervals $[-2.07, 2.12]$ m and $[-0.95, 1.01]$ m, respectively along the x and z directions.

N_k:	51	Weighting rule:	Inverse
MAE($x_\Omega, \widehat{x}_\Omega$):	0.453 m	MAE($z_\Omega, \widehat{z}_\Omega$):	0.260 m
Mean($x_\Omega - \widehat{x}_\Omega$):	0.028 m	Mean($z_\Omega - \widehat{z}_\Omega$):	0.027 m
StdDev($x_\Omega - \widehat{x}_\Omega$):	1.07 m	StdDev($z_\Omega - \widehat{z}_\Omega$):	0.50 m

Table 5.6: Railway bridge - Accuracy of damage localization along the x and z directions using $\mathcal{L}_2$ with $N_h = 4$.

Comparison with former research activity. The railway bridge test case considered in this Section offers an opportunity for comparison of the deep metric learning strategy described so far with

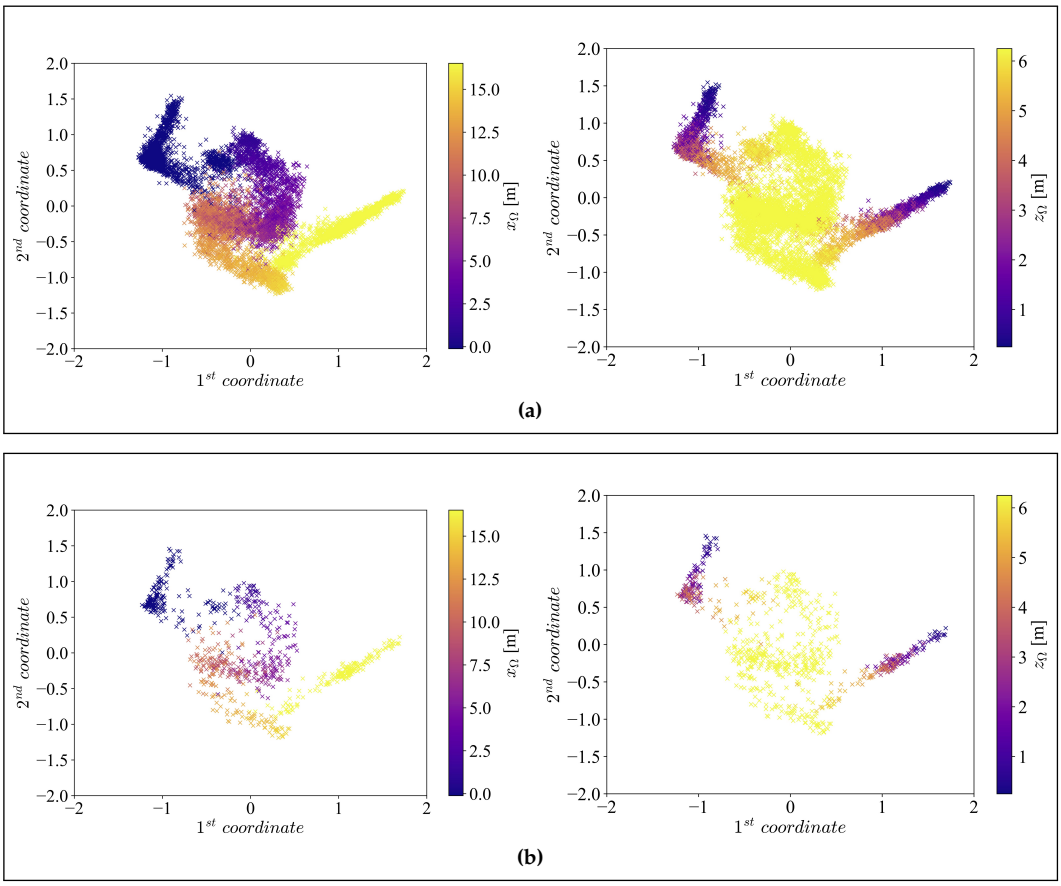


Figure 5.22: Railway bridge - Multidimensional scaling onto two dimensions of the $\mathrm{NN}_{\mathbf{h}}$ embeddings using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$: (a) training data $\{\mathbf{h}_i = (h_1, h_2, h_3, h_4)_i^\top\}_{i=1}^{N_{\text{train}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{train}}}$ and (right) z-direction $\{z_{\Omega i}\}_{i=1}^{N_{\text{train}}}$; (b) testing data $\{\mathbf{h}_j = (h_1, h_2, h_3, h_4)_j^\top\}_{j=1}^{N_{\text{test}}}$ against the correct damage position along the (left) x-direction $\{x_{\Omega j}\}_{j=1}^{N_{\text{test}}}$ and (right) z-direction $\{z_{\Omega j}\}_{j=1}^{N_{\text{test}}}$.

an alternative real-time damage localization strategy. Within this latter, the damage localization task is approached from a classification perspective. Specifically, following the setup described in Sec. 4.1.2, raw vibration recordings are mapped onto discrete target labels, each associated with predefined damage scenarios of interest. The mapping is learned by minimizing the categorical cross-entropy between the predicted and target class labels, according to Eq. (4.8). This strategy shares several similarities with the proposed method: (*i*) it relies on the simulation-based paradigm of SHM; (*ii*) reduced-order modeling is employed to alleviate the computational burden associated with the generation of training datasets; and (*iii*) it exploits a supervised DL-based mapping of raw vibration recordings to assess the location of damage. The reader is referred to [94], for a complete description.

In the context of the same railway bridge case study, the proposed strategy significantly improves the damage localization outcomes in several ways: (*i*) the accuracy with which damage is located is markedly refined compared to the former work,

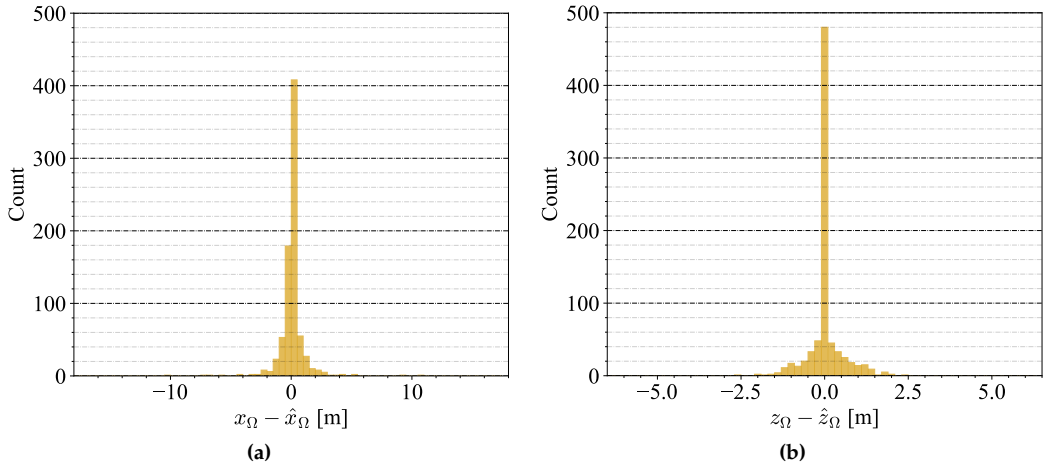


Figure 5.23: Railway bridge - Distribution of prediction errors along the (a) x-direction and (b) z-direction, using $\mathcal{L}_{\mathbf{h}}^{(2)}$ with $N_h = 4$.

which accounted for only six damage scenarios associated with relatively large damageable regions; (ii) damage localization now supports a varying magnitude δ of the stiffness reduction taking place within the damage region, whereas δ was held constant in the former work; and (iii) moving from a classification perspective to a metric-based one, the proposed method eliminates the need of defining target classes in advance, allowing damage to be identified anywhere in the structure.

5.5 CONCLUSIONS

Within a simulation-based SHM framework, in this Chapter, we have proposed a method for damage localization that enables the learning of a smooth mapping of vibration data onto a low-dimensional space. This mapping relies on a DNN trained through a supervised pairwise approach within a Siamese architecture. The proposed setting allows for a damage-sensitive manifold capable of representing distances in the physical space and yielding easily interpretable results. Furthermore, the use of convolutional layers allows the automatic selection and extraction of damage-sensitive features from raw vibration recordings. The procedure takes advantage of a preliminary offline phase involving the use of physics-based numerical models and reduced order modeling, to address the lack of experimental data while populating the training dataset. During the online phase, the location of damage is assessed through a k-NN regression performed on the learned low-dimensional space.

The proposed method can be employed to locate damage at various scales, eliminating the need of defining multiple target damage classes and tuning highly sensitive hyperparameters,

such as distance thresholds and decision boundaries. Its effectiveness has been assessed on an L-shaped cantilever beam and a portal frame railway bridge, demonstrating improved damage localization compared to an alternative real-time strategy. The study has investigated the use of three alternative loss functions for learning the mapping, while also varying the size of the low-dimensional feature space. In all considered cases, as long as the low-dimensional mapping features a sufficient representation capability to describe the variability of the monitored system in terms of structural health, the damage localization performance has shown remarkable accuracy. This result is achieved in the presence of varying operational conditions and measurement noise — an aspect that is judged particularly valuable in view of application to real-world large-scale structures.

The best damage localization outcomes have been achieved by learning the target mapping through the minimization of the Square-Square loss function, equipped with the Euclidean metric. The resulting mapping has shown robustness against measurement noise, varying operational conditions characterized by time-varying and moving loads, and different levels of damage severity. Another point of strength lies in the resilience to variations in the assumed size of the damaged region — an aspect that has been considered to assess the potential impact of inaccuracies in modeling the most likely damage mechanisms occurring in the monitored structure.

Concerning the limitations, the primary concern is associated with the uncertainties linked to the modeling assumptions. An inaccurately specified numerical model of the monitored structure could bias the training data, posing a risk to the generalization capabilities of the damage identification strategy upon deployment. This represents a typical example of data-to-methodology gap, wherein a methodology well-trained in a simulated environment may experience a significant drop in performance when applied to the target structure — an issue that we have not investigated and that could potentially be mitigated using domain adaptation strategies.

This page intentionally left blank.

6 MULTI-FIDELITY SURROGATE MODELING

Part of the research presented in this Chapter has been published as

[165] *Torzoni et al., A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks,* Mechanical Systems and Signal Processing, *Vol. 197, 2023, p. 110376.*

[168] *Torzoni et al., A Deep Neural Network, Multi-fidelity Surrogate Model Approach for Bayesian Model Updating in SHM,* European Workshop on Structural Health Monitoring, Springer International Publishing, *2023, pp. 1076–1086.*

[169] *Torzoni et al., Health Monitoring of Civil Structures: A MCMC Approach Based on a Multi-Fidelity Deep Neural Network Surrogate,* Computer Sciences & Mathematics Forum, *Vol. 2, No. 1, 2022, p. 16.*

[340] *Torzoni et al., A Multi-Fidelity Deep Neural Network Approach to Structural Health Monitoring,* Engineering Proceedings, *Vol. 27, No. 1, 2022, p. 60.*

This Chapter is devoted to development of a *non-intrusive* MF surrogate modeling strategy, designed to approximate sensor recordings under varying damage and operational conditions. ROMs and DNNs are employed to reduce the computational cost of supervised training while ensuring the accuracy of the approximated signals in terms of damage sensitivity. The proposed framework can be useful, for example, for model-updating purposes, as demonstrated in this Chapter, or for inexpensively generating training datasets, as shown in Chapter 7.

The proposed MF surrogate model relies on a composition of DNNs, comprising an LF and an HF component, and is therefore termed *MF-DNN*. The *LF-DNN* is a fully-connected model used to replicate sensor recordings in the undamaged condition, while the *HF-DNN* is an LSTM employed to adaptively refine the LF approximation with damage effects. This *multi-level architecture* is trained sequentially by leveraging datasets of varying fidelity. The LF-DNN is trained on a large dataset of *LF structural responses* in the absence of damage, generated through a computationally efficient *LF-ROM* relying on the reduced basis method. Simultaneously, the HF-DNN is trained on a smaller dataset of *HF structural responses* that account for potential degradation processes. This dataset is generated using an *HF-FE* model, considered the most accurate description for possible damage scenarios. The decision to employ different models for varying levels of approximation aligns with our MF framework: the LF-DNN captures the major trends of the problem, while the HF-DNN enhances the LF approximation by exploiting the correlation between the two fidelity levels. Indeed, the LSTM model behind the HF-DNN leverages the temporal structure of the LF approximation to fully capitalize on the limited HF data. This configuration enables superior approximation capabilities compared to *single-fidelity SMs*, without increasing the computational cost of supervised training.

The novelty of the proposed methodology lies in the tailored adaptation of the MF-DNN surrogate modeling framework for SHM purposes. The MF-DNN-SM is well-suited for high-dimensional problems and benefits from large LF training

datasets. It learns both linear and nonlinear correlations adaptively without the need for prior information, can handle the approximation of strongly discontinuous functions, and provides real-time predictions for new input parameters. Moreover, by simulating the effect of damage on the structural response with the HF-FE model alone, we overcome the potential challenge of retaining a large number of POMs in the LF-ROM to accurately describe damages. This approach allows us to cheaply generate a significant amount of training data relevant to the undamaged condition, thereby making the use of ROMs extremely effective to speed up the offline training phase. Finally, there is no need to update the LF component, as when a deterioration in structural health is detected, the MF surrogate can be updated by adjusting only its HF component.

The Chapter is organized as follows. In Sec. 6.1, we specify the proposed surrogate modeling framework in term of the low/high-fidelity training datasets and the low/high-fidelity SM components. In Sec. 6.2, we investigate the effectiveness and efficiency of the proposed MF-DNN-SM using the same test cases considered in the previous Chapter*. The achieved approximation capabilities are shown to outperform those provided by a single-fidelity SM counterpart. Finally, in Sec. 6.3, we assess the capability of the MF-DNN to be exploited within an MCMC-based strategy, aimed at updating SHM parameters conditioned on sensor recordings. Here, the MF-DNN-SM is employed to systematically speed up the evaluation of the likelihood function of the MCMC sampler, while the deep metric learning setting proposed in Chapter 5 is exploited to provide an effective initialization of the MCMC simulation. Conclusions are finally drawn in Sec. 6.4.

*The same case studies are also employed in the following Chapters to compare the different objectives reached in this Thesis.

6.1 MULTI-FIDELITY DEEP NEURAL NET SURROGATE MODEL

The MF-DNN surrogate model $\mathrm{NN}_{\mathrm{MF}}$ is designed to map input parameters onto time series data. It leverages an LF component and an HF component, *sequentially trained*, and respectively denoted as $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$. The resulting SM reads as:

$$\mathrm{NN}_{\mathrm{MF}}(\boldsymbol{\mu}^{\mathrm{HF}}) = \mathrm{NN}_{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}) \circ \mathrm{NN}_{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}), \tag{6.1}$$

where $\circ$ stands for function composition. In Eq. (6.1), $\boldsymbol{\mu}^{\mathrm{LF}} \in \mathbb{R}^{N_{\mathrm{par}}^{\mathrm{LF}}}$ includes $N_{\mathrm{par}}^{\mathrm{LF}} \in \mathbb{N}$ parameters describing the *operational conditions*, while $\boldsymbol{\mu}^{\mathrm{HF}} \in \mathbb{R}^{N_{\mathrm{par}}^{\mathrm{HF}}}$, with $\boldsymbol{\mu}^{\mathrm{LF}} \subseteq \boldsymbol{\mu}^{\mathrm{HF}}$, includes $N_{\mathrm{par}}^{\mathrm{HF}} \in \mathbb{N}$ parameters referring to both *operational and damage conditions*.

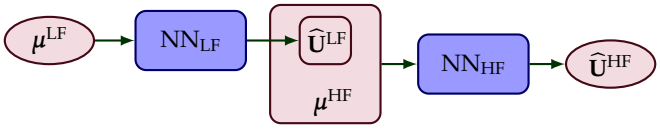


Figure 6.1: MF-DNN surrogate model. Red nodes denote the input/output quantities, while blue nodes refer to the learnable components of the surrogate model. Hat variables denote neural network approximations.

$\mathrm{NN}_{\mathrm{LF}}$ serves to approximate LF vibration recordings in the absence of damage, denoted as $\mathbf{U}^{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}) = [\mathbf{u}_1^{\mathrm{LF}}, \dots, \mathbf{u}_{N_u}^{\mathrm{LF}}] \in \mathbb{R}^{L \times N_u}$ and consisting of N_u series with L measurements equally spaced in time, for any given set of *LF input parameters* $\boldsymbol{\mu}^{\mathrm{LF}}$. Subsequently, $\mathrm{NN}_{\mathrm{HF}}$ takes the *HF input parameters* $\boldsymbol{\mu}^{\mathrm{HF}}$ and the *LF approximation* $\widehat{\mathbf{U}}^{\mathrm{LF}} \in \mathbb{R}^{L \times N_u}$ from $\mathrm{NN}_{\mathrm{LF}}$ to provide its *HF enrichment* $\widehat{\mathbf{U}}^{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}) \in \mathbb{R}^{L \times N_u}$, embedding the effects from damage to approximate the target HF vibration recordings $\mathbf{U}^{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}) = [\mathbf{u}_1^{\mathrm{HF}}, \dots, \mathbf{u}_{N_u}^{\mathrm{HF}}] \in \mathbb{R}^{L \times N_u}$ (see Fig. 6.1 for a schematic).

The $\mathrm{NN}_{\mathrm{MF}}$ multi-level architecture is chosen over an "all-in-one scheme" that relies on a single DNN to approximate both LF and HF vibration data. This choice stems from the higher complexity involved in training the latter, which necessitates a multi-objective optimization to simultaneously learn the LF and HF components. For instance, an all-in-one approach, combining the contributions from two fidelity levels through a weighted sum in the loss function, is exploited in [147]. However, we highlight the risk of potential numerical instabilities induced by large discrepancies between the two contributions and the need to tune the relative mixing ratio. Moreover, in [149], an all-in-one architecture (termed "intermediate model") trained by jointly exploiting LF and HF data was also shown to underperform alternative MF models tuned through a sequential learning of the LF approximation and the HF refinement.

147. Meng *et al.*, *A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems.* 2020

149. Guo *et al.*, *Multi-fidelity regression using artificial neural networks: Efficient approximation of parameter-dependent output quantities.* 2022

In the remainder of this Section, we specify the content of the LF and HF datasets in Sec. 6.1.1, and we discuss the $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ components, along with the technical aspects related to training and evaluation in Sec. 6.1.2.

6.1.1 Low/High fidelity datasets

The MF-DNN is trained on synthetic data, generated through physics-based models. Specifically, we use *LF and HF datasets*, denoted as $\mathcal{D}_{\mathrm{LF}}$ and $\mathcal{D}_{\mathrm{HF}}$, which collect simulated structural response data in the absence or presence of damage, respectively. While LF data pertain to the baseline condition, HF data must account for potential degradation processes at variance with the formerly identified damaged condition. Even though this distinction identifies a *bi-fidelity* setting, we highlight that the correlation between $\mathcal{D}_{\mathrm{LF}}$ and $\mathcal{D}_{\mathrm{HF}}$ changes with the health state of the structure, and the MF-DNN should adapt to any identified

damage condition as necessary. Without loss of generality, we refer to the initial monitoring phase of an undamaged condition.

The chosen physics-based numerical models consists of an LF-ROM relying on the reduced basis method and an HF-FE model. The HF-FE model underlying $\mathcal{D}_{\text{HF}}$ relies on Problem (3.1) to simulate structural responses under varying damage and operational conditions. Following the same setup employed in Chapter 5, damage is simulated by reducing the material stiffness by a magnitude δ, within a subdomain $\Omega_{\mathbf{y}}$ identified by the spatial coordinates $\mathbf{y}$ of its center of mass. Both $\mathbf{y}$ and δ are included in the vector of HF input parameters $\boldsymbol{\mu}^{\text{HF}}$. The LF-ROM underlying $\mathcal{D}_{\text{LF}}$ instead refers to Problem (3.18). Besides not accounting for the presence of damage, the damping term is also disregarded in the LF model to better validate the MF-DNN performance on more different LF and HF structural responses.

Solving Problem (3.18) and Problem (3.1), displacement values are collected in $(0, T)$ as $\mathbf{V}_{\text{LF}}(\boldsymbol{\mu}^{\text{LF}}) = [\mathbf{W}\widehat{\mathbf{x}}_1, \dots, \mathbf{W}\widehat{\mathbf{x}}_L] \in \mathbb{R}^{N_{\text{FE}} \times L}$ and as $\mathbf{V}_{\text{HF}}(\boldsymbol{\mu}^{\text{HF}}) = [\mathbf{x}_1, \dots, \mathbf{x}_L] \in \mathbb{R}^{N_{\text{FE}} \times L}$, respectively . The corresponding vibration recordings $\mathbf{U}^{\text{LF}}(\boldsymbol{\mu}^{\text{LF}})$ and $\mathbf{U}^{\text{HF}}(\boldsymbol{\mu}^{\text{HF}})$ are then extracted using Eq. (3.23) for $\mathbf{V}_{\text{LF}}$ and $\mathbf{V}_{\text{HF}}$, respectively.

By following the procedure outlined in Sec. 3.2, the datasets $\mathcal{D}_{\text{LF}}$ and $\mathcal{D}_{\text{HF}}$ are populated by taking $\boldsymbol{\mu}^{\text{LF}}$ and $\boldsymbol{\mu}^{\text{HF}}$ as uniformly distributed, and then sampling them via Latin hypercube rule. The number of samples corresponds to $N_{\text{train}}^{\text{LF}}$ and $N_{\text{train}}^{\text{HF}}$, with $N_{\text{train}}^{\text{LF}} > N_{\text{train}}^{\text{HF}}$, instances collected in $\mathcal{D}_{\text{LF}}$ and $\mathcal{D}_{\text{HF}}$ respectively, similar to Eq. (4.3):

$$\mathcal{D}_{\text{LF}} = \{(\boldsymbol{\mu}_{i_{\text{LF}}}^{\text{LF}}, \mathbf{U}_{i_{\text{LF}}}^{\text{LF}})\}_{i_{\text{LF}}=1}^{N_{\text{train}}^{\text{LF}}}, \tag{6.2}$$

$$\mathcal{D}_{\text{HF}} = \{(\boldsymbol{\mu}_{i_{\text{HF}}}^{\text{HF}}, \mathbf{U}_{i_{\text{HF}}}^{\text{HF}})\}_{i_{\text{HF}}=1}^{N_{\text{train}}^{\text{HF}}}. \tag{6.3}$$

Here, the i_{LF}-th sampling of $\boldsymbol{\mu}^{\text{LF}}$ and the i_{HF}-th sampling of $\boldsymbol{\mu}^{\text{HF}}$, with $i_{\text{LF}} = 1, \dots, N_{\text{train}}^{\text{LF}}$ and $i_{\text{HF}} = 1, \dots, N_{\text{train}}^{\text{HF}}$, are labeled by the corresponding vibration recordings $\mathbf{U}_{i_{\text{LF}}}^{\text{LF}}(\boldsymbol{\mu}_{i_{\text{LF}}}^{\text{LF}})$ and $\mathbf{U}_{i_{\text{HF}}}^{\text{HF}}(\boldsymbol{\mu}_{i_{\text{HF}}}^{\text{HF}})$, respectively. In the following, indices i_{LF} and i_{HF} will be dropped for ease of notation, unless explicitly required.

The numerical models exploited to generate $\mathcal{D}_{\text{LF}}$ and $\mathcal{D}_{\text{HF}}$ differ in their approximation qualities and computational costs. While, in our case, the HF model is required to accurately replicate structural damage, this is not restrictive, and the proposed MF-DNN modeling strategy can easily be adapted to handle data with varying characteristics. Additionally, we assume that the sampling frequency f_{s} and the number N_u of monitored DOFs are consistent for both fidelity levels, although there are no restrictions in this regard.

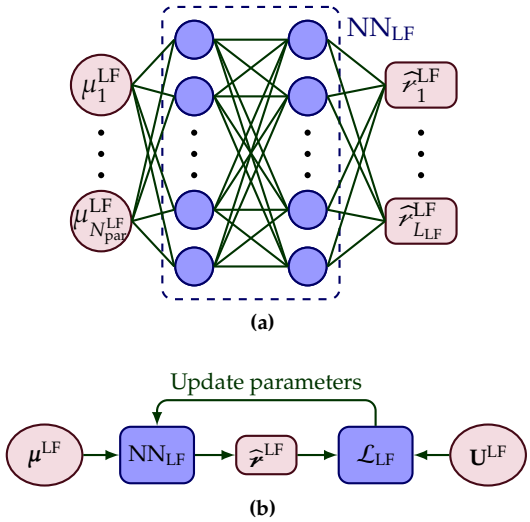


Figure 6.2: MF-DNN surrogate model - NN$_{\text{LF}}$ component: (a) scheme of the fully-connected model; (b) training flowchart. Red nodes refer to the input/output quantities, while blue nodes denote the relevant computational blocks. Hat variables denote quantities obtained from neural network approximations.

6.1.2 Low/High fidelity surrogate components

The MF-DNN-SM consists of a *fully-connected* LF-DNN NN$_{\text{LF}}$, which mimics sensor recordings in the undamaged condition, and an LSTM HF-DNN NN$_{\text{HF}}$, which adaptively refines the LF approximation with the effect of damage.

The LF component NN$_{\text{LF}}$: $\mathbb{R}^{N_{\text{par}}^{\text{LF}}} \mapsto \mathbb{R}^{L_{\text{LF}}}$, $\boldsymbol{\mu}^{\text{LF}} \mapsto \boldsymbol{r}^{\text{LF}}$, is a regressor employed to map the LF input parameters $\boldsymbol{\mu}^{\text{LF}}$ onto a set of $L_{\text{LF}} \in \mathbb{N}$ *weighted POD coefficients* $\boldsymbol{r}^{\text{LF}}$, see Fig. 6.2a, that encodes $\mathbf{U}^{\text{LF}}$ as:

$$\boldsymbol{r}^{\text{LF}}(\boldsymbol{\mu}^{\text{LF}}) = \boldsymbol{w} \odot \mathbf{Y}_{\text{LF}}^{\top}\texttt{vec}[\mathbf{U}^{\text{LF}}]. \tag{6.4}$$

In Eq. (6.4), $\mathbf{Y}_{\text{LF}} \in \mathbb{R}^{L_{\texttt{vec}} \times L_{\text{LF}}}$, with $L_{\texttt{vec}} = LN_u$, is a matrix of L_{LF} POMs to reduce the dimensionality of $\texttt{vec}[\mathbf{U}^{\text{LF}}]$ from $L_{\texttt{vec}}$ to L_{LF}, and therefore, the number of learnable parameters in NN$_{\text{LF}}$. In particular, $\mathbf{Y}$ gathers the first L_{LF} POMs, with $L_{\text{LF}} \ll L_{\texttt{vec}}$, of the vectorized LF instances collected as $\mathbf{S}_{\text{LF}} = [\texttt{vec}[\mathbf{U}_1^{\text{LF}}], \dots, \texttt{vec}[\mathbf{U}_{N_{\text{train}}^{\text{LF}}}^{\text{LF}}]] \in \mathbb{R}^{L_{\texttt{vec}} \times N_{\text{train}}^{\text{LF}}}$. The Hadamard element-wise product $\odot : (\mathbb{R}, \mathbb{R}) \mapsto \mathbb{R}$ scales the corresponding POD coefficients by the vector $\boldsymbol{w} \in \mathbb{R}^{L_{\text{LF}}}$. This vector is employed to weight the relative importance of the POMs, acting as an inductive bias that reflects the decreasing information represented by each POM, as further clarified in Sec. 6.2.

To train NN$_{\text{LF}}$, the vibration recordings in $\mathcal{D}_{\text{LF}}$ are preprocessed to generate a *dedicated training dataset*, as follows:

$$\mathcal{D}_{\text{LF}}^{\texttt{vec}} = \{(\boldsymbol{\mu}_{i_{\text{LF}}}^{\text{LF}}, \boldsymbol{r}_{i_{\text{LF}}}^{\text{LF}} = \boldsymbol{w} \odot \mathbf{Y}_{\text{LF}}^{\top}\texttt{vec}[\mathbf{U}_{i_{\text{LF}}}^{\text{LF}}])\}_{i_{\text{LF}}=1}^{N_{\text{train}}^{\text{LF}}}, \tag{6.5}$$

so that the LF input parameters $\boldsymbol{\mu}^{\text{LF}}$ and the corresponding weighted POD coefficients can be used to tune the learnable

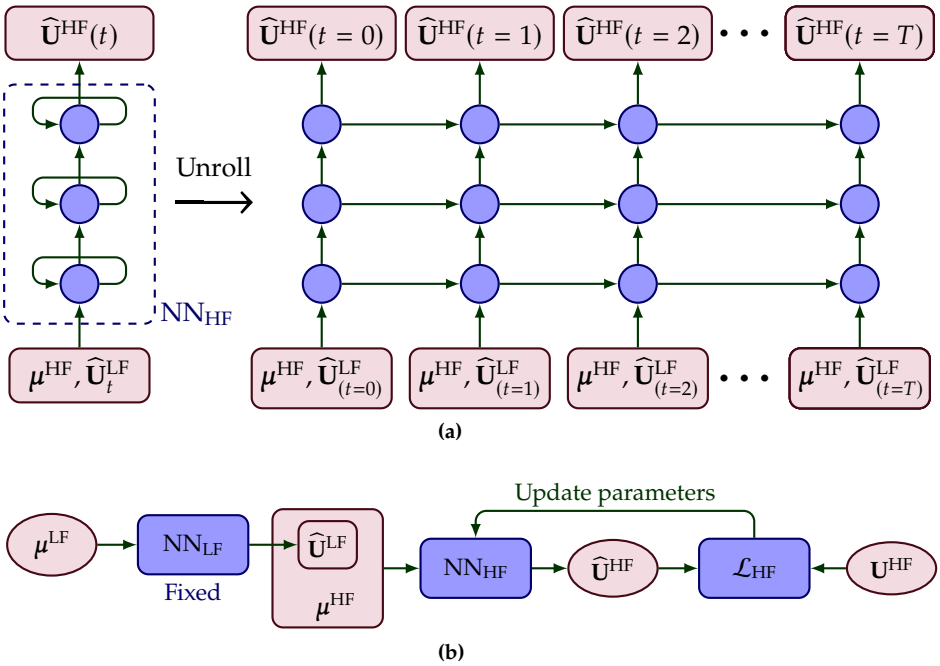


Figure 6.3: MF-DNN surrogate model - $\mathrm{NN}_{\mathrm{HF}}$ component: (a) scheme of the recurrent model; (b) training flowchart. Red nodes refer to the input/output quantities, while blue nodes denote the relevant computational blocks. Hat variables denote quantities obtained from neural network approximations.

weights $\boldsymbol{\Omega}_{\mathrm{LF}}$ parametrizing $\mathrm{NN}_{\mathrm{LF}}$. This is achieved by minimizing the following loss function:

$$\mathcal{L}_{\mathrm{LF}}(\boldsymbol{\Omega}_{\mathrm{LF}}, \mathcal{D}^{\mathrm{vec}}_{\mathrm{LF}}) = \frac{1}{N^{\mathrm{LF}}_{\mathrm{train}}} \sum_{i_{\mathrm{LF}}=1}^{N^{\mathrm{LF}}_{\mathrm{train}}} \|\boldsymbol{r}^{\mathrm{LF}}_{i_{\mathrm{LF}}} - \mathrm{NN}_{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}_{i_{\mathrm{LF}}})\|_2^2. \tag{6.6}$$

The relevant training process is outlined in Fig. 6.2b.

For any given set of LF input parameters $\boldsymbol{\mu}^{\mathrm{LF}}$, the trained $\mathrm{NN}_{\mathrm{LF}}$ provides an approximated reconstruction of $\mathbf{U}^{\mathrm{LF}}$, as:

$$\widehat{\mathbf{U}}^{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}) = \mathtt{vec}^{-1}_{L\times N_u}[\mathbf{Y}_{\mathrm{LF}}(\frac{1}{\boldsymbol{w}} \odot \mathrm{NN}_{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}))], \tag{6.7}$$

where $\widehat{\boldsymbol{r}}^{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}}) = \mathrm{NN}_{\mathrm{LF}}(\boldsymbol{\mu}^{\mathrm{LF}})$ is the approximated counterpart of $\boldsymbol{r}$, and $\mathtt{vec}^{-1}_{N_R\times N_C} : \mathbb{R}^{N_R N_C} \to \mathbb{R}^{N_R\times N_C}$ denotes the inverse of the vectorization operation.

The HF component $\mathrm{NN}_{\mathrm{HF}} : (\mathbb{R}^{N^{\mathrm{HF}}_{\mathrm{par}}}, \mathbb{R}^{N_u}) \mapsto \mathbb{R}^{N_u}$, $(\boldsymbol{\mu}^{\mathrm{HF}}, \widehat{\mathbf{U}}^{\mathrm{LF}}_t) \mapsto \mathbf{U}^{\mathrm{HF}}(t)$, is a *regressor built upon the LSTM model*, used to exploit the *time correlation* between the two fidelity levels. At each time step, $\mathrm{NN}_{\mathrm{HF}}$ takes the HF parameters $\boldsymbol{\mu}^{\mathrm{HF}}$, and the LF approximation $\widehat{\mathbf{U}}^{\mathrm{LF}}_t$, enriching it with the effects of damage and structural damping to provide $\widehat{\mathbf{U}}^{\mathrm{HF}}(t)$. See also Fig. 6.3a.

The choice to employ different models for the varying levels of approximation is related to our MF framework, characterized by a *hierarchy of fidelity levels*: HF data are scarce due to their cost, while LF data are readily obtainable at the expense of

reduced accuracy. In this context, $\mathrm{NN}_{\mathrm{LF}}$ provides information on the major trends of the problem, while $\mathrm{NN}_{\mathrm{HF}}$ enhances the LF approximation by exploiting the correlation between the two fidelity levels. Additionally, the choice to employ an LSTM model for $\mathrm{NN}_{\mathrm{HF}}$ is motivated by the possibility to exploit the temporal structure of $\widehat{\mathbf{U}}^{\mathrm{LF}}$, provided as input to $\mathrm{NN}_{\mathrm{HF}}$, and fully capitalize on the limited available HF data. In contrast, $\mathrm{NN}_{\mathrm{LF}}$ lacks access to any input time series related to the target structural response, making a fully-connected model that maps onto POD coefficients a more effective strategy. We also highlight that the training of a fully-connected model is easily parallelizable, making $\mathrm{NN}_{\mathrm{LF}}$ suitable for learning on very large LF datasets. On the other hand, the backpropagation through time algorithm required by an LSTM model would impose a non-negligible computational burden if training $\mathrm{NN}_{\mathrm{HF}}$ on large HF datasets.

The weights $\boldsymbol{\Omega}_{\mathrm{HF}}$ parametrizing $\mathrm{NN}_{\mathrm{HF}}$ are tuned by minimizing the following loss function:

$$\mathcal{L}_{\mathrm{HF}}(\boldsymbol{\Omega}_{\mathrm{HF}}, \mathcal{D}_{\mathrm{HF}}) = \frac{1}{N_{\mathrm{train}}^{\mathrm{HF}}} \frac{1}{N_{\mathrm{C}}} \sum_{i_{\mathrm{HF}}=1}^{N_{\mathrm{train}}^{\mathrm{HF}}} \sum_{n_{\mathrm{C}}=1}^{N_{\mathrm{C}}} \| \mathtt{vec}[\mathbf{U}^{\mathrm{HF}}_{i_{\mathrm{HF}},(n_{\mathrm{C}}:n_{\mathrm{C}}+L_{\mathrm{C}})} - \mathrm{NN}_{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}_{i_{\mathrm{HF}}}, \widehat{\mathbf{U}}^{\mathrm{LF}}_{(n_{\mathrm{C}}:n_{\mathrm{C}}+L_{\mathrm{C}})}(\boldsymbol{\mu}^{\mathrm{LF}}_{i_{\mathrm{HF}}}))] \|_1. \tag{6.8}$$

Here, $L_{\mathrm{C}} \in \mathbb{N}$ is the chunk length for the backpropagation through time algorithm discussed in Sec. 4.2.6, resulting in a total of $N_{\mathrm{C}} = L - L_{\mathrm{C}} + 1$ *overlapping chunks* obtained from each recording. The relevant training process is carried out as outlined in Fig. 6.3b, with the weights $\boldsymbol{\Omega}_{\mathrm{LF}}$ of $\mathrm{NN}_{\mathrm{LF}}$ kept fixed.

In Eq. (6.8), it is worth noting that we employ an MAE-like objective function to adjust the prediction accuracy of $\mathrm{NN}_{\mathrm{HF}}$, which has proven superior over an MSE-like one. We attribute this improvement to its reduced sensitivity to atypical training data instances, such as vibration recordings obtained by exciting the structure close to resonance frequencies. We stress that the performance of $\mathrm{NN}_{\mathrm{HF}}$ remains satisfactory even when using an MSE-like objective function. However, this is suboptimal compared to the chosen objective function, especially when the LF component $\mathrm{NN}_{\mathrm{LF}}$ is not exploited. This observation is particularly relevant for the *single-fidelity model* $\mathrm{NN}_{\mathrm{HF}}^{\mathrm{only}}$ considered for comparison in the next Section, which is identical to $\mathrm{NN}_{\mathrm{HF}}$ but lacks the input channels providing $\widehat{\mathbf{U}}^{\mathrm{LF}}$.

For any given set of HF input parameters $\boldsymbol{\mu}^{\mathrm{HF}}$, the trained $\mathrm{NN}_{\mathrm{HF}}$ approximates $\mathbf{U}^{\mathrm{HF}}$, as follows:

$$\widehat{\mathbf{U}}^{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}, t) = \mathrm{NN}_{\mathrm{HF}}(\boldsymbol{\mu}^{\mathrm{HF}}, \widehat{\mathbf{U}}^{\mathrm{LF}}_{t}(\boldsymbol{\mu}^{\mathrm{LF}})). \tag{6.9}$$

Algorithm 6.1 Multi-fidelity deep neural network surrogate. Preliminary offline phase: algorithmic description

Input: parametrization of the physics-based models
POD tolerance ϵ_σ for LF-ROM
training options for the LF component $\mathrm{NN}_{\mathrm{LF}}$
training options for the HF component $\mathrm{NN}_{\mathrm{HF}}$

1: set up the LF-FOM of the structure
2: derive the LF-ROM from LF-FOM analyses
3: use the LF-ROM to populate the LF dataset $\mathcal{D}_{\mathrm{LF}}$
4: assemble the snapshot matrix $\mathbf{S}_{\mathrm{LF}}$ from $\mathcal{D}_{\mathrm{LF}}$
5: compute matrix $\mathbf{Y}_{\mathrm{LF}}$ via SVD of $\mathbf{S}_{\mathrm{LF}}$
6: pre-process $\mathcal{D}_{\mathrm{LF}}$ to derive $\mathcal{D}_{\mathrm{LF}}^{\mathsf{vec}}$
7: train $\mathrm{NN}_{\mathrm{LF}}$ on $\mathcal{D}_{\mathrm{LF}}^{\mathsf{vec}}$
8: test $\mathrm{NN}_{\mathrm{LF}}$ against LF-FOM data
9: set up the HF-FE of the structure
10: use the HF-FE to populate the HF dataset $\mathcal{D}_{\mathrm{HF}}$
11: train $\mathrm{NN}_{\mathrm{HF}}$ on $\mathcal{D}_{\mathrm{HF}}$
12: test $\mathrm{NN}_{\mathrm{HF}}$ against HF-FE data
13: **return** trained LF component $\mathrm{NN}_{\mathrm{LF}}$,
14: trained HF component $\mathrm{NN}_{\mathrm{HF}}$

The algorithmic description of the procedures and computations characterizing the preliminary offline phase of the proposed strategy is reported in Algorithm 6.1. Once trained, $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ can be jointly exploited to evaluate $\mathrm{NN}_{\mathrm{MF}}$ for new input parameters with negligible computational time.

6.2 Validation and Verification of Surrogate Modeling

The approximation capabilities of $\mathrm{NN}_{\mathrm{MF}}$ are assessed against two different systems: the L-shaped cantilever beam and the railway bridge already considered in Chapter 5. The former case study aims to illustrate the use of the proposed MF surrogate modeling framework in a simple SHM scenario. Additionally, it aims to analyze the impact of various choices of the $\boldsymbol{w}$ weighting vector on the performance of $\mathrm{NN}_{\mathrm{LF}}$, providing insights for its selection. The second case study is adopted to discuss the computational benefits of the strategy and to highlight possible issues regarding its generalization capabilities for systems characterized by a remarkable structural/geometrical complexity.

The software and hardware setup used to implement and evaluate the HF-FE models and the LF-ROMs, as well as the architectures of $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$, are the same as those exploited in the previous Chapter. The implementation details of $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ are provided in Appendix B.

6.2.1 SURROGATE OF AN L-SHAPED CANTILEVER BEAM

The first test case we consider is the L-shaped beam, previously addressed in Sec. 5.3. We avoid to fully re-describe the considered setting and instead focus only on the different aspects.

The applied harmonic loading is parametrized with $Q \in [1, 5]$ kPa and $f_q \in [10, 60]$ Hz. Displacement time histories $\mathbf{U}^{\mathrm{LF}}$ and $\mathbf{U}^{\mathrm{HF}}$ are recorded by means of $N_u = 8$ DOFs over a time interval $(0, T = 1\text{ s})$ with an acquisition frequency of $f_{\mathrm{s}} = 200$ Hz, to mimic the sensing system depicted in Fig. 5.2.

The HF-FE model features $N_{\mathrm{FE}} = 4659$ DOFs. The damping matrix is assembled according to the Rayleigh's model to achieve a 5% damping ratio on the first four structural modes. Damage is simulated by reducing the material stiffness within $\Omega_{\mathbf{y}}$, whose location is parametrized through $\mathbf{y} = (x_\Omega, y_\Omega)^\top$, with either x_Ω or y_Ω varying in the range $[0.15, 3.85]$ m. The damage level is set to $\delta = 25\%$, and it remains constant within $(0, T)$. Thus, the vector of HF input parameters is $\boldsymbol{\mu}^{\mathrm{HF}} = (Q, f_q, x_\Omega, y_\Omega)^\top$.

The matrix $\mathbf{W}$ underlying the LF-ROM is derived from a snapshot matrix $\mathbf{S}$, assembled by integrating over $(0, T)$ an LF-FOM that does not account for the presence of damage and damping, for $N_P = 200$ varied values of the LF input parameters $\boldsymbol{\mu}^{\mathrm{LF}} = (Q, f_q)^\top$. Prescribing a tolerance $\epsilon_\sigma = 10^{-3}$ in Eq. (3.16), the order of the LF-ROM is set to $N_{\mathrm{RB}} = 14$. The computing time required to assemble each LF instance accordingly decreases from 4.90 s to 0.19 s, of which only 0.024 s is needed to advance the solution in time, entailing a speed-up of about 26 times.

For the case at hand, $\mathcal{D}_{\mathrm{LF}}$ is populated with $N^{\mathrm{LF}}_{\mathrm{train}} = 10,000$ LF-ROM data instances collected for training $\mathrm{NN}_{\mathrm{LF}}$, while $\mathcal{D}_{\mathrm{HF}}$ is populated with an additional $N^{\mathrm{HF}}_{\mathrm{train}} = 1000$ HF-FE data instances collected for training $\mathrm{NN}_{\mathrm{HF}}$. Regarding the dimensionality reduction of the LF recordings, the matrix $\mathbf{Y}_{\mathrm{LF}}$ accounts for $L_{\mathrm{LF}} = 104$ POMs, selected by performing an SVD on $\mathbf{S}_{\mathrm{LF}}$ with a chosen tolerance of $\epsilon_\sigma = 10^{-3}$. The testing data used to assess the performance of $\mathrm{NN}_{\mathrm{LF}}$ and $\mathrm{NN}_{\mathrm{HF}}$ consist of $N^{\mathrm{LF}}_{\mathrm{test}} = 40$ LF and $N^{\mathrm{HF}}_{\mathrm{test}} = 40$ HF data instances, respectively.

Concerning the weighting vector $\boldsymbol{w}$, we argue that appropriately weighting the POMs of $\mathbf{S}_{\mathrm{LF}}$, prioritizing those associated with higher information content, is more effective than using a uniform $\boldsymbol{w}$. To provide some insights on such a selection of $\boldsymbol{w}$, we systematically assess the performance of $\mathrm{NN}_{\mathrm{LF}}$ for three alternative choices of $\boldsymbol{w}$. The results are reported in Tab. 6.1, considering: (i) a flat $\boldsymbol{w}$ with all unit entries; (ii) a linearly decaying $\boldsymbol{w}$ with entries $w_l = 1 - \frac{0.8}{L_{\mathrm{LF}}-1}(l-1)$, $l = 1, \ldots, L_{\mathrm{LF}}$; and (iii) a $\boldsymbol{w}$ with entries given by the singular values of $\mathbf{S}_{\mathrm{LF}}$.

The $\mathrm{NN}_{\mathrm{LF}}$ performance is measured in terms of average and

	$\mathrm{PCC}_{\mathrm{avg}}\{u_1(t), u_4(t), u_8(t)\}$	$\mathrm{PCC}_{\mathrm{min}}\{u_1(t), u_4(t), u_8(t)\}$	$\mathrm{MAE}\{u_1(t), u_4(t), u_8(t)\}$
$\boldsymbol{w} = \mathbf{1}$	$\{0.994, 0.995, 0.999\}$	$\{0.954, 0.965, 0.992\}$	$\{0.319, 2.179, 2.737\} \cdot 10^{-6}$
$\boldsymbol{w} = (\{1 - \frac{0.8}{L_{\mathrm{LF}}-1}(l-1)\}_{l=1}^{L_{\mathrm{LF}}})^\top$	$\{0.998, 0.998, 0.999\}$	$\{0.992, 0.986, 0.994\}$	$\{0.196, 1.481, 2.185\} \cdot 10^{-6}$
$\boldsymbol{w} = (\{\sigma_l(\mathbf{S}_{\mathrm{LF}})\}_{l=1}^{L_{\mathrm{LF}}})^\top$	$\{0.884, 0.971, 0.980\}$	$\{0.628, 0.911, 0.901\}$	$\{1.193, 5.012, 8.821\} \cdot 10^{-6}$

Table 6.1: L-shaped cantilever beam - $\mathrm{NN}_{\mathrm{LF}}$ performance indicators for the LF testing signals at the $u_1(t)$, $u_4(t)$, and $u_8(t)$ monitored DOFs. Three choices of $\boldsymbol{w}$ are considered: flat; linearly decaying; ruled by the singular values of $\mathbf{S}_{\mathrm{LF}}$.

minimum *Pearson correlation coefficients** ($\mathrm{PCC}_{\mathrm{avg}}$ and $\mathrm{PCC}_{\mathrm{min}}$) between the predicted and ground truth LF sensor recordings $u_1(t)$, $u_4(t)$, and $u_8(t)$, and in terms of MAE between the same LF signals. While $\mathrm{PCC}_{\mathrm{avg}}$ and $\mathrm{PCC}_{\mathrm{min}}$ offer dimensionless measures of the model's *goodness-of-fit*, MAE provides an *absolute error indicator*. For more details, see e.g., [341].

*The Pearson correlation coefficient quantifies the linear correlation between two sets of data. It is computed by dividing the covariance of the two variables by the product of their standard deviations.

341. Ritter *et al.*, *Performance evaluation of hydrological models: Statistical significance for reducing subjectivity in goodness-of-fit assessments*. 2013

Although the best results are obtained by linearly weighing the POMs by their relative importance, the performance of $\mathrm{NN}_{\mathrm{LF}}$ is more than satisfactory even when using the flat $\boldsymbol{w}$. The attained values of $\mathrm{PCC}_{\mathrm{min}}$ above 0.95 and those of $\mathrm{PCC}_{\mathrm{avg}}$ even close to 1 suggest a strong, positive association between predicted and target LF signals. It is interesting to note that $\mathrm{PCC}_{\mathrm{min}}$ also increases when the first POD-modes are prioritized. This may appear surprising and suggest that ϵ_σ can be increased without significantly spoiling $\widehat{\mathbf{U}}^{\mathrm{LF}}$. Weighting instead the POD coefficients of $\mathbf{S}_{\mathrm{LF}}$ by its singular values results in a degradation of all the considered metrics, with achieved values of $\mathrm{PCC}_{\mathrm{min}}$ even below of 0.65, and values of MAE four times higher than those obtained with the linearly decaying $\boldsymbol{w}$. Therefore, we argue that the beneficial effect of such weighting is compromised if high POMs are excessively penalized. Based on this analysis, the linearly decaying $\boldsymbol{w}$ is selected for the following experiments.

Some exemplary signals provided by $\mathrm{NN}_{\mathrm{MF}}$ for the monitored DOFs $u_1(t)$, $u_4(t)$, and $u_8(t)$ are presented in Fig. 6.4. Each row of plots display the ground truth HF displacement time series alongside the LF and HF approximations under different operational and damage conditions. From a qualitative perspective, the HF approximation consistently aligns with the target signal in both low and high-frequency components. Despite the discrepancy between the LF approximations and the target HF signals, the proposed MF setting allows $\mathrm{NN}_{\mathrm{HF}}$ to effectively exploit the time correlation between the two fidelity levels to properly enrich the LF approximation.

For comparison, the same results are also reported in Fig. 6.5 as obtained with the single-fidelity counterpart of $\mathrm{NN}_{\mathrm{MF}}$, termed $\mathrm{NN}_{\mathrm{HF}}^{\mathrm{only}}$. This model is identical to $\mathrm{NN}_{\mathrm{HF}}$ except that the input channels providing $\widehat{\mathbf{U}}^{\mathrm{LF}}$ are removed. The obtained results demonstrate that preventing $\mathrm{NN}_{\mathrm{HF}}^{\mathrm{only}}$ from exploiting the correlation between HF and LF recordings has a significantly negative impact on its approximation capabilities. Despite the substantial

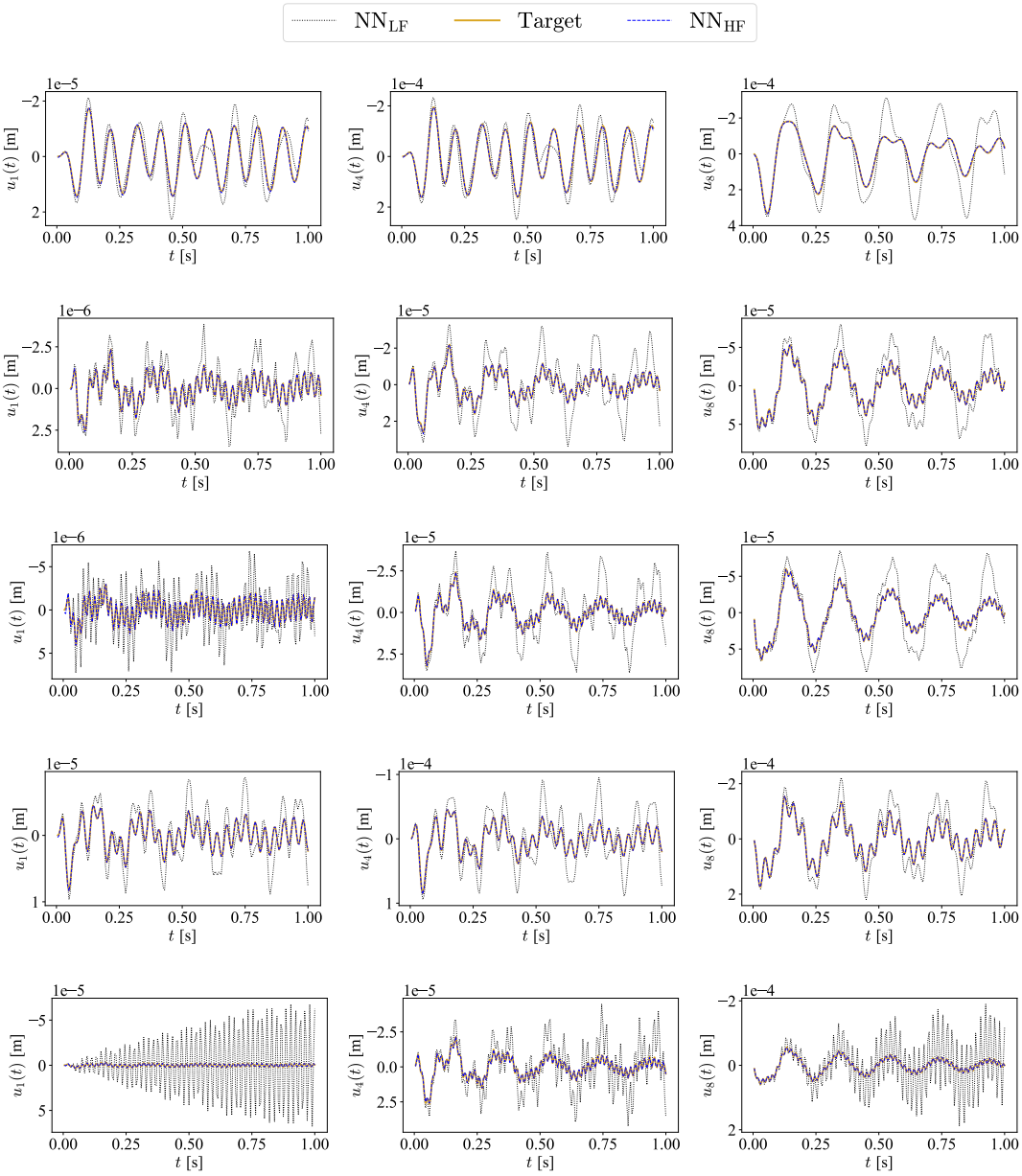


Figure 6.4: L-shaped cantilever beam - Exemplary comparisons of $u_1(t)$, $u_4(t)$, and $u_8(t)$ displacement time histories provided by $\mathrm{NN_{LF}}$ and $\mathrm{NN_{HF}}$ with the target HF signals. Results in the five rows are obtained for the following values of input parameters: $Q = \{2.37, 2.61, 4.56, 4.71, 4.90\}$ kPa; $f_q = \{10.32, 35.10, 46.93, 26.23, 53.26\}$ Hz; $x_\Omega = \{1.70, 3.18, 3.85, 3.85, 3.85\}$ m; $y_\Omega = \{0.15, 0.15, 0.76, 1.28, 1.93\}$ m.

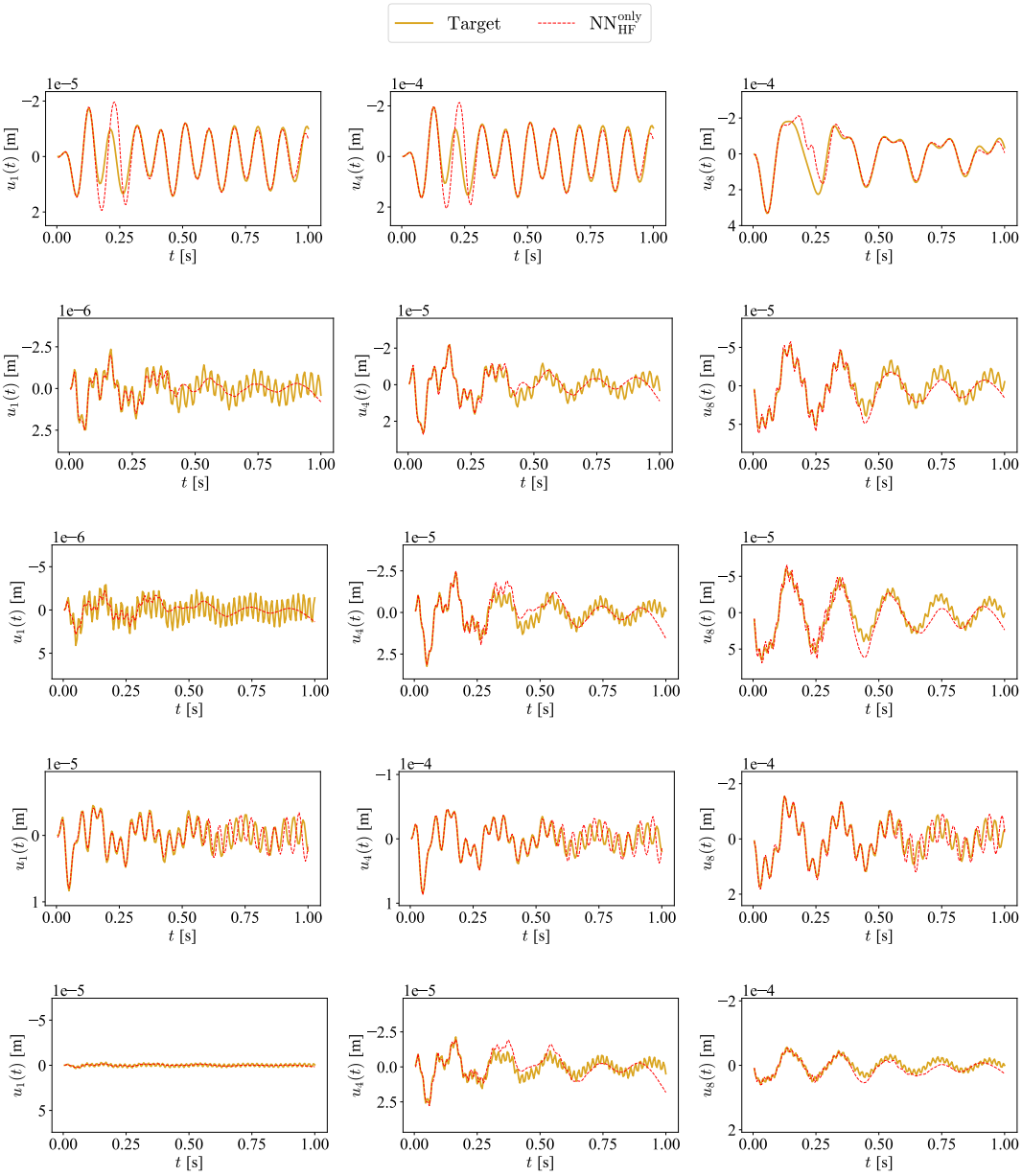


Figure 6.5: L-shaped cantilever beam - Exemplary comparisons of $u_1(t)$, $u_4(t)$, and $u_8(t)$ displacement time histories provided by $\mathrm{NN}_{\mathrm{HF}}^{\mathrm{only}}$ with the target HF signals. Results in the five rows are obtained for the following values of input parameters: $Q = \{2.57, 2.61, 4.56, 4.71, 4.90\}$ kPa; $f_q = \{10.32, 35.10, 46.93, 26.23, 53.26\}$ Hz; $x_\Omega = \{1.70, 3.18, 3.85, 3.85, 3.85\}$ m; $y_\Omega = \{0.15, 0.15, 0.76, 1.28, 1.93\}$ m.

	$\text{PCC}_{\text{avg}}\{u_1(t), u_4(t), u_8(t)\}$	$\text{PCC}_{\text{min}}\{u_1(t), u_4(t), u_8(t)\}$	$\text{MAE}\{u_1(t), u_4(t), u_8(t)\}$
NN_{MF}	$\{0.997, 0.999, 0.999\}$	$\{0.987, 0.993, 0.994\}$	$\{0.804, 6.130, 9.055\} \cdot 10^{-7}$
$\text{NN}_{\text{HF}}^{\text{only}}$	$\{0.677, 0.823, 0.908\}$	$\{0.215, 0.467, 0.685\}$	$\{0.773, 5.894, 10.815\} \cdot 10^{-6}$

Table 6.2: L-shaped cantilever beam - NN_{HF} performance indicators for the HF testing signals at the $u_1(t)$, $u_4(t)$, $u_8(t)$ monitored DOFs.

discrepancy between LF approximations and target HF signals, it is interesting to note how NN_{LF} seems to play a crucial role in maintaining correct trajectories without accumulating errors over the long-time range, especially in the case of signals rich in high-frequency components.

To quantify the performance gain provided by the LF component, we compute the PCC_{avg}, PCC_{min} and MAE indicators between predicted and ground truth HF signals for the same sample channels. The obtained values are reported in Tab. 6.2, as obtained by considering both NN_{HF} and $\text{NN}_{\text{HF}}^{\text{only}}$. When $\text{NN}_{\text{HF}}^{\text{only}}$ is used instead of NN_{HF}, significant degradations are observed in all the considered performance measures. For instance, the attained values of PCC_{min}, which are all above 0.95 in the MF setting, drop to 0.215, 0.467, and 0.685, respectively, for DOFs $u_1(t)$, $u_4(t)$, and $u_8(t)$, while the corresponding MAE values increase by one order of magnitude. This loss of accuracy is attributed to the fact that $\text{NN}_{\text{HF}}^{\text{only}}$ not only needs to enrich the LF signals with the effects of damage and damping, but it also must entirely reproduce the HF *input-output behavior*.

In Fig. 6.6, we present the *learning curves*, illustrating the evolution of the loss function over the training and validation sets for NN_{LF} and NN_{HF}. The initial training epochs show the most significant improvements in both cases. The irregular spikes observed during the early training stages are attributed to the stochastic nature of the optimization algorithm employed. The good agreement between the training and validation curves suggests that the models have not encountered overfitting issues.

Finally, we compare the computational times required by NN_{MF} and the HF-FE model to generate the $N_{\text{test}}^{\text{HF}} = 40$ testing instances. The CPU time required by the FE solver amounts to 196 s, while, the GPU time required by NN_{MF} reduces to 0.054 s. Therefore, NN_{MF} run considerably faster than real-time, with a speed-up of about 3630 times with respect to the HF-FE model.

6.2.2 SURROGATE OF A RAILWAY BRIDGE

This second case study involves the railway bridge previously considered in Sec. 5.4. Again, we refrain from fully re-describing the considered setting and instead focus only on the differences.

LF and HF displacement time histories are recorded using the $N_u = 10$ sensors deployed as depicted in Fig. 6.7 over a time interval $(0, T = 1.5\text{ s})$ with a sampling frequency of $f_s = 400$ Hz.

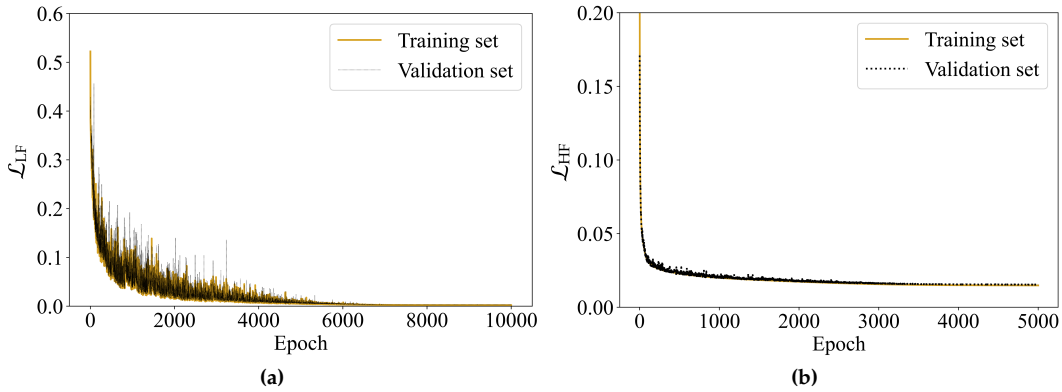


Figure 6.6: L-shaped cantilever beam - Learning curves displaying the loss function evolution on the training and validation sets for (a) NN_{LF} and (b) NN_{HF}.

The HF-FE model features $N_{\text{FE}} = 17,292$ DOFs. The Rayleigh damping matrix accounts for a 5% damping ratio on the first four structural modes. Damage is modeled by a localized stiffness reduction that can occur anywhere over the two frame walls and the deck, within subdomains $\Omega_{\mathbf{y}}$ featuring a different layout in the two cases (see Fig. 6.7). The position of $\Omega_{\mathbf{y}}$ is parametrized through $\mathbf{y} = (x_\Omega, z_\Omega)^\top$, with either x_Ω or z_Ω varying in the ranges $[-0.115, 16.515]$ m and $[0.4, 6.25]$ m, respectively. In this case, the damage level can take values within $\delta \in [20\%, 50\%]$, and it remains constant while a train travels across the bridge. To summarize, the vector of HF input parameters is $\boldsymbol{\mu}^{\text{HF}} = (v, m_A, x_\Omega, z_\Omega, \delta)^\top$.

The LF-ROM is obtained from a snapshot matrix $\mathbf{S}$, assembled through $N_P = 200$ evaluations over $(0, T)$ of an LF-FOM not accounting for the presence of damage and damping, for different values of LF input parameters $\boldsymbol{\mu}^{\text{LF}} = (v, m_A)^\top$. By setting the POD error tolerance to $\epsilon_\sigma = 10^{-3}$, the order of the LF-ROM is set to $N_{\text{RB}} = 312$. The computing time required to assemble each LF instance accordingly decreases from 87.00 s to 11.80 s, entailing a speed-up of approximately 7 times.

The dataset $\mathcal{D}_{\text{LF}}$ is populated with $N_{\text{train}}^{\text{LF}} = 5000$ LF-ROM data instances, while $\mathcal{D}_{\text{HF}}$ is populated with $N_{\text{train}}^{\text{HF}} = 500$ HF-FE data instances. $\mathcal{D}_{\text{LF}}^{\text{vec}}$ is then derived from $\mathcal{D}_{\text{LF}}$ by means of a projection matrix $\mathbf{Y}_{\text{LF}}$ accounting for $L_{\text{LF}} = 68$ POMs, selected by performing an SVD on $\mathbf{S}_{\text{LF}}$ with a chosen tolerance of $\epsilon_\sigma = 10^{-3}$. The testing data used to assess the performance of NN_{LF} and NN_{HF} consist of $N_{\text{test}}^{\text{LF}} = 40$ LF and $N_{\text{test}}^{\text{HF}} = 40$ HF data instances, respectively. We remark that NN_{LF} and NN_{HF} are identical to those considered in Sec. 6.2.1, except for the size of the input/output layers, which is problem-specific.

Some results obtained with NN_{MF} are reported in Fig. 6.8 for sensors $u_2(t)$, $u_4(t)$, $u_5(t)$, $u_7(t)$, and $u_{10}(t)$. Each column of plots provides the ground truth HF displacement time series together

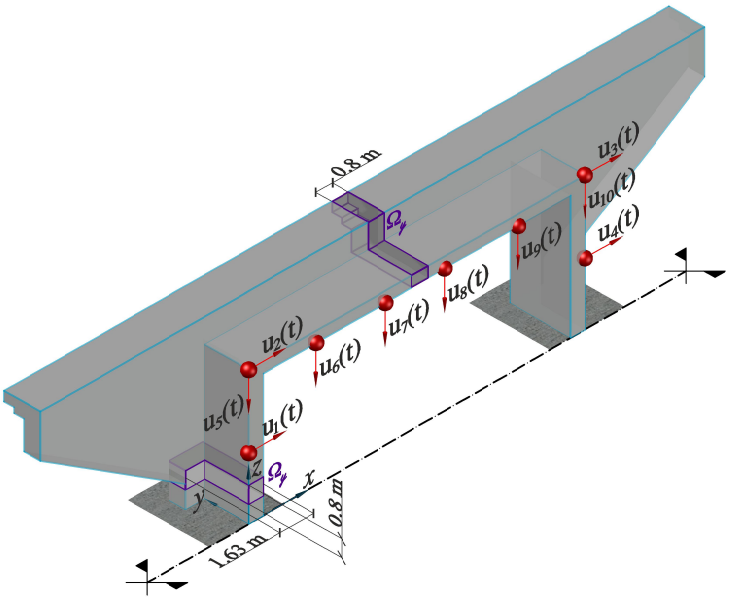


Figure 6.7: Railway bridge - Details of synthetic recordings related to displacements $u_1(t), \ldots, u_{10}(t)$, and damageable region Ω_y.

	$\text{PCC}_{\text{avg}}\{u_4(t), u_5(t), u_7(t)\}$	$\text{PCC}_{\text{min}}\{u_4(t), u_5(t), u_7(t)\}$	$\text{MAE}\{u_4(t), u_5(t), u_7(t)\}$
NN_{MF}	$\{0.999, 0.999, 0.999\}$	$\{0.999, 0.996, 0.999\}$	$\{0.553, 0.559, 7.434\} \cdot 10^{-7}$
$\text{NN}_{\text{HF}}^{\text{only}}$	$\{0.483, 0.544, 0.641\}$	$\{0.163, 0.497, 0.526\}$	$\{0.173, 0.448, 4.374\} \cdot 10^{-5}$
NN_{LF} (wrt LF)	$\{0.999, 0.999, 0.999\}$	$\{0.999, 0.999, 0.999\}$	$\{0.400, 0.446, 3.224\} \cdot 10^{-7}$

Table 6.3: Railway bridge - NN_{LF} and NN_{HF} performance indicators for the HF and LF testing signals at the $u_4(t)$, $u_5(t)$ and $u_7(t)$ monitored DOFs.

with the LF and HF approximations under different operational and damage conditions. From a qualitative point of view, NN_{MF} effectively handles the nonlinear correlations between LF and HF data, even in the presence of strongly discontinuous trajectories, as observed in the time histories of $u_5(t)$ and $u_{10}(t)$.

The same results are also reported in Fig. 6.9, as obtained with $\text{NN}_{\text{HF}}^{\text{only}}$. According to what reported for the L-shaped beam case, a severe performance degradation is observed when NN_{LF} is removed. The approximated time histories are highly unreliable for most of the time, even though the considered time histories in this case do not exhibit significant high-frequency content. The approximations are only piece-wise accurate, with sharp variations simultaneously affecting all channels. Besides the limited amount of HF data instances exploited to train $\text{NN}_{\text{HF}}^{\text{only}}$, this behavior seems to be triggered by the occurrence of sharp discontinuities in the $u_5(t)$ and $u_{10}(t)$ time histories.

The performance indicators associated with NN_{HF}, $\text{NN}_{\text{HF}}^{\text{only}}$, and NN_{LF} are reported in Tab. 6.3, as obtained for $u_4(t)$, $u_5(t)$, and $u_7(t)$. The obtained results agree with what observed in the previous case study, with significant drops in the attained values of PCC_{avg} and PCC_{min}, and an increase of almost two orders of magnitude in the MAE values when NN_{LF} is removed. On the other hand, by exploiting the proposed MF setting, the

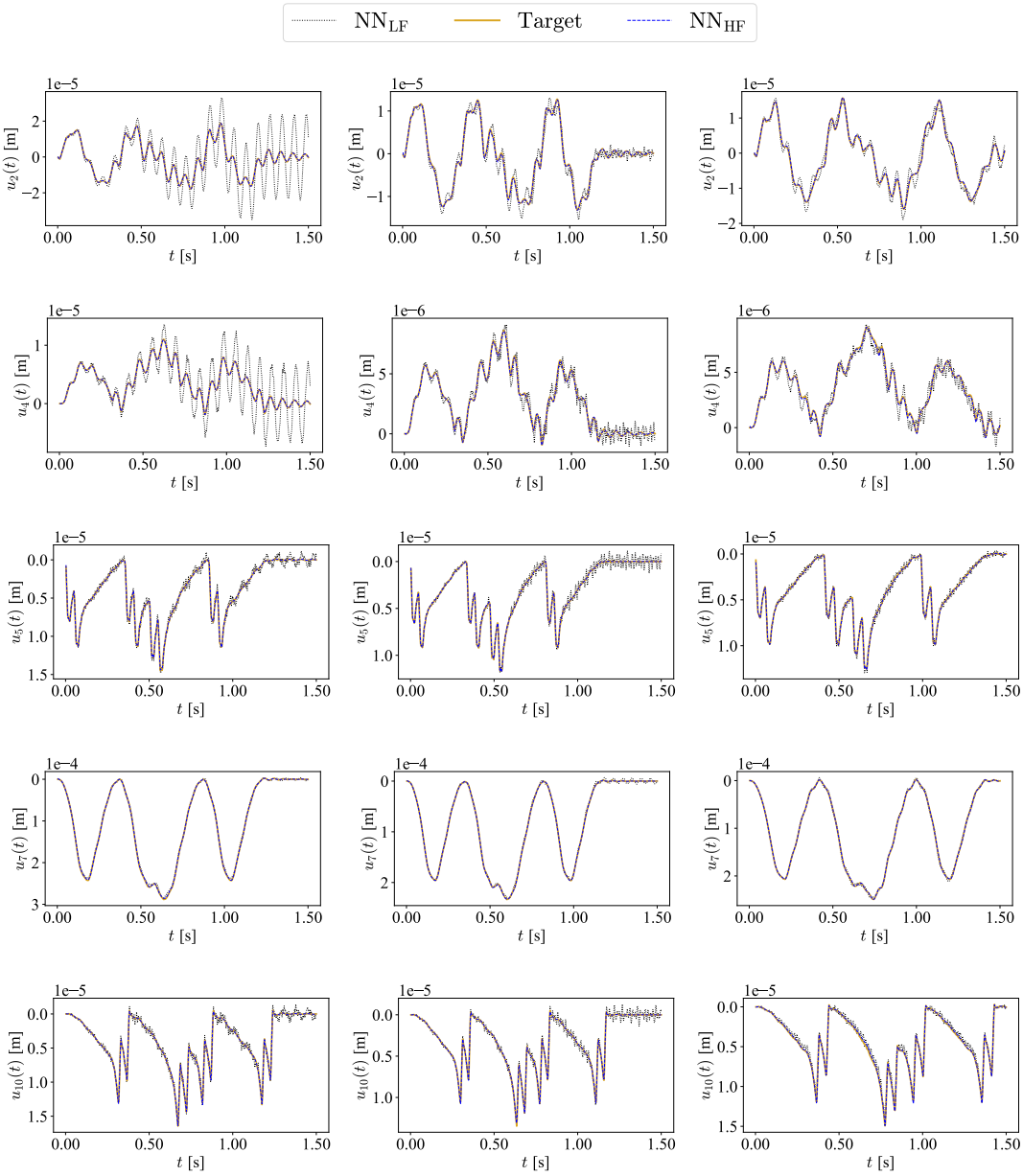


Figure 6.8: Railway bridge - Exemplary comparisons of $u_2(t)$, $u_4(t)$, $u_5(t)$, $u_7(t)$ and $u_{10}(t)$ displacement time histories provided by $\mathrm{NN_{LF}}$ and $\mathrm{NN_{HF}}$ with the target HF signals. Results in the three columns are obtained for the following values of input parameters: $\upsilon = \{192.1, 203.48, 166.59\}$ km/h; $m_A = \{20.66, 16.85, 17.95\}$ ton; $x_\Omega = \{5.97, 4.95, 16.515\}$ m; $z_\Omega = \{6.25, 6.25.2.46\}$ m; $\delta = \{0.23, 0.42, 0.30\}$%.

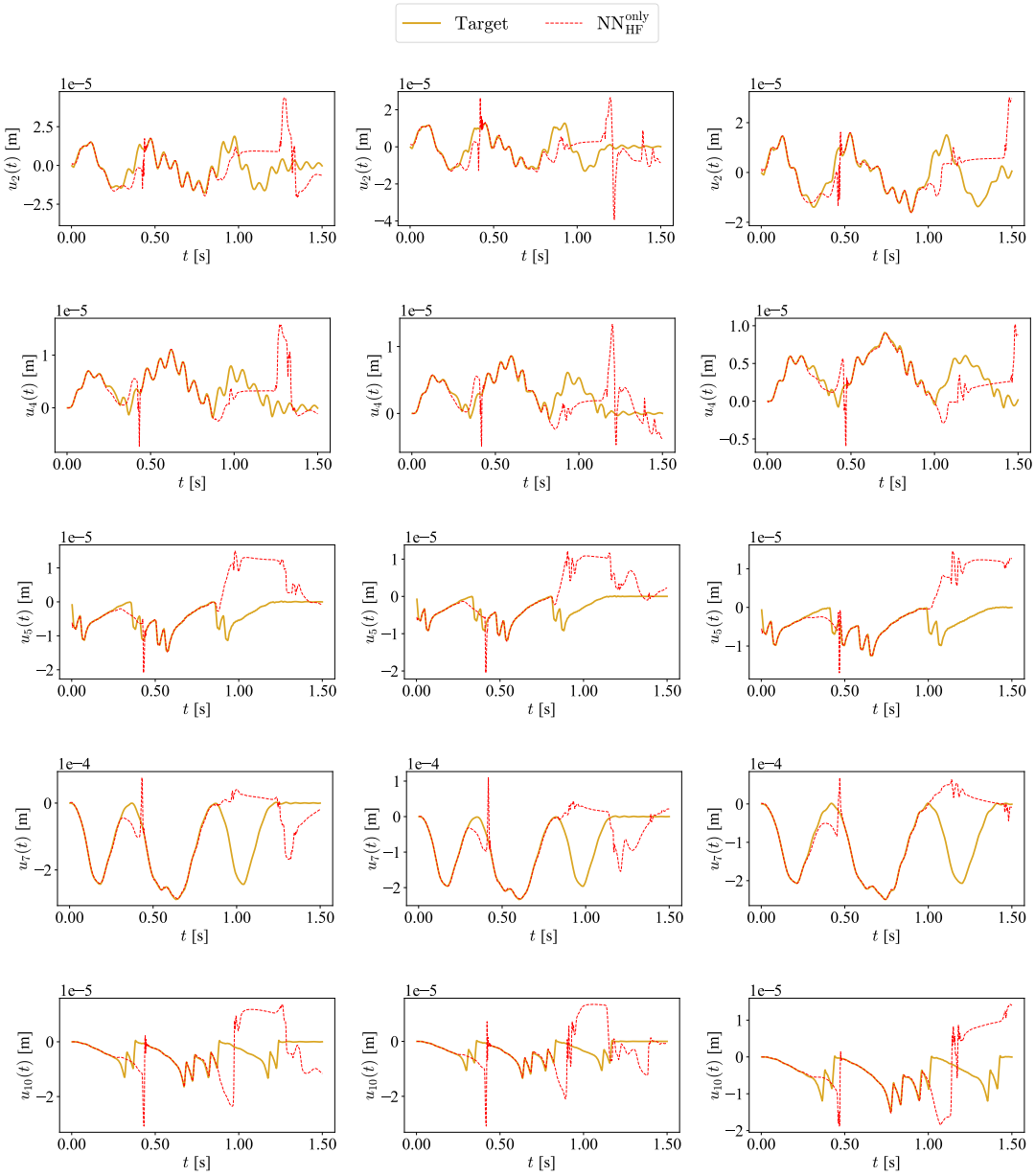


Figure 6.9: Railway bridge - Exemplary comparisons of $u_2(t)$, $u_4(t)$, $u_5(t)$, $u_7(t)$ and $u_{10}(t)$ displacement time histories provided by NN_{HF}^{only} with the target HF signals. Results in the three columns are obtained for values of input parameters: $v = \{192.1, 203.48, 166.59\}$ km/h; $m_A = \{20.66, 16.85, 17.95\}$ ton; $x_\Omega = \{5.97, 4.95, 16.515\}$ m; $z_\Omega = \{6.25, 6.25.2.46\}$ m; $\delta = \{0.23, 0.42, 0.30\}$%.

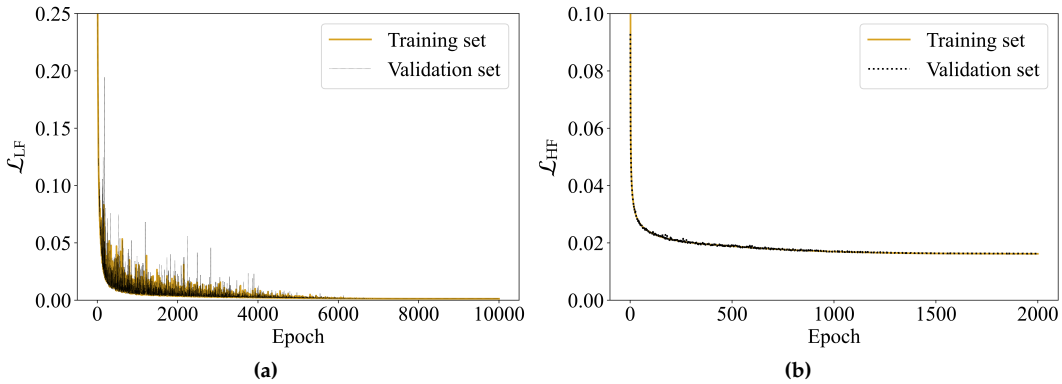


Figure 6.10: Railway bridge - Learning curves displaying the loss function evolution on the training and validation sets for (a) NN_{LF} and (b) NN_{HF}.

obtained values of PCC_{avg} and PCC_{min} are both extremely close to 1, indicating that the nonlinear correlations between LF and HF data are perfectly assimilated by NN_{MF}.

The learning curves showing the evolution of $\mathcal{L}_{\text{LF}}$ and $\mathcal{L}_{\text{HF}}$ over the training and validation sets are reported in Fig. 6.10. Similar to the L-shaped beam case, the most significant improvements occur during the early training stages, with increasingly subtle performance enhancements as the number of training epochs increases. Again, we can appreciate satisfactory generalization capabilities for both $\mathcal{L}_{\text{LF}}$ and $\mathcal{L}_{\text{HF}}$.

Regarding the computational times required by NN_{MF} to generate the $N_{\text{test}}^{\text{HF}} = 40$ testing instances, the HF-FE solver takes 3480 s, while NN_{MF} only takes 0.125 s, resulting in a speed-up of about $27,840$ times compared to the HF numerical model.

6.3 SIMULATED IDENTIFICATION: L-SHAPED BEAM

To discuss the capability provided by the proposed MF scheme in identifying damage patterns, we employ an MF-DNN-SM within an MCMC-based model updating framework. The considered SHM scenario involves the L-shaped beam presented in Sec. 6.2.1. The sought parameters are represented by the vector $\boldsymbol{\theta} = (\theta_f, \theta_Q, \theta_\Omega)^\top$, where the target position of $\Omega_{\mathbf{y}}$ — determined by the coordinates of its center of mass — is encoded as an abscissa $\theta_\Omega \in [0.15, 7.55]$ m running along the axis of the structure. MCMC sampling is employed to approximate the posterior PDF $p(\boldsymbol{\theta}|\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$, conditioned on a batch of $N_{\text{obs}} \in \mathbb{N}$ measurements $\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}} = [\mathbf{U}^{\text{EXP}}_1, \dots, \mathbf{U}^{\text{EXP}}_{N_{\text{obs}}}] \in \mathbb{R}^{N_{\text{obs}} \times L \times N_u}$, under assumed steady operational and damage conditions.

To provide θ_f and θ_Q with an informative prior for the MCMC simulation, a *preliminary identification* of these parameters is addressed through a *deep metric learning* feature extractor NN_{OC} — with the subscript OC standing for "operational con-

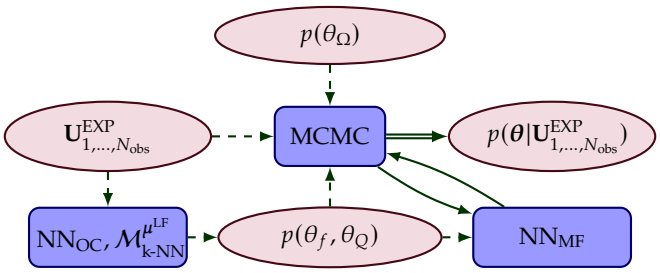


Figure 6.11: Flowchart of the employed SHM strategy. Red nodes refer to the input/output quantities, while blue nodes denote the relevant computational blocks.

ditions" — and a *k-NN regressor* $\mathcal{M}^{\boldsymbol{\mu}^{\text{LF}}}_{\text{k-NN}}$, following the strategy presented in Sec. 5.1. The obtained prior not only constraints the Bayesian inference of $\boldsymbol{\theta}$ within a plausible range, but also serves to initialize the Markov chains in high probability regions.

Fig. 6.11 provides a high-level overview of the adopted SHM strategy. The identification of operational conditions occurs in the lower-left node (NN_{OC}, $\mathcal{M}^{\boldsymbol{\mu}^{\text{LF}}}_{\text{k-NN}}$) and is detailed in Sec. 6.3.1. The MCMC-based Bayesian inference of $\boldsymbol{\theta}$ involves using NN_{MF} to systematically speed up the evaluation of a likelihood function, as explained in Sec. 6.3.2. Finally, the obtained results are reported and discussed in Sec. 6.3.3.

6.3.1 Data-driven preliminary identification

The preliminary identification of θ_f and θ_Q relies on *mapping vibration recordings onto a low-dimensional feature space*. Once learned, this mapping is deployed to extract data features from previously unseen data, feeding a *k-NN regressor* to provide informed predictions for the parameters underlying $\boldsymbol{\mu}^{\text{LF}}$.

The low-dimensional mapping is driven by a *fully-connected DL model* $\text{NN}_{\text{OC}} : \mathbb{R}^{L_{\text{OC}}} \mapsto \mathbb{R}^{N_h}$, $\boldsymbol{\varkappa}^{\text{HF}} \mapsto \mathbf{h}$, aimed at providing the feature representation $\mathbf{h}(\boldsymbol{\mu}^{\text{LF}})$ in a low-dimensional space of size $N_h \in \mathbb{N}$ for a set of $L_{\text{OC}} \in \mathbb{N}$ POD coefficients $\boldsymbol{\varkappa}^{\text{HF}}$. These coefficients encode $\mathbf{U}^{\text{HF}}(\boldsymbol{\mu}^{\text{HF}})$ similarly to Eq. (6.4), as follows:

$$\boldsymbol{\varkappa}^{\text{HF}}(\boldsymbol{\mu}^{\text{HF}})) = \mathbf{Y}_{\text{OC}}^{\top}\mathsf{vec}[\mathbf{U}^{\text{HF}}]. \tag{6.10}$$

In Eq. (6.10), $\mathbf{Y}_{\text{OC}} \in \mathbb{R}^{L_{\text{vec}} \times L_{\text{OC}}}$ is a matrix gathering the first L_{OC} POD modes of the vectorized HF data instances, collected as $\mathbf{S}_{\text{HF}} = [\mathsf{vec}[\mathbf{U}^{\text{HF}}_1], \dots, \mathsf{vec}[\mathbf{U}^{\text{HF}}_{N^{\text{HF}}_{\text{train}}}]] \in \mathbb{R}^{L_{\text{vec}} \times N^{\text{HF}}_{\text{train}}}$, with $L_{\text{OC}} \ll L_{\text{vec}}$, which is useful to perform a *linear dimensionality reduction* of the HF recordings. Given the layout of the L-shaped beam and the employed parametrization for $\boldsymbol{\mu}^{\text{HF}}$, we expect the first POMs of $\mathbf{S}_{\text{HF}}$ to be highly sensitive to the parameters underlying $\boldsymbol{\mu}^{\text{LF}}$. Reducing the dimensionality of $\mathbf{U}^{\text{HF}}(\boldsymbol{\mu}^{\text{HF}})$ in this way thus helps to limit the number of learnable parameters ruling NN_{OC} without compromising the identification of $\boldsymbol{\mu}^{\text{LF}}$.

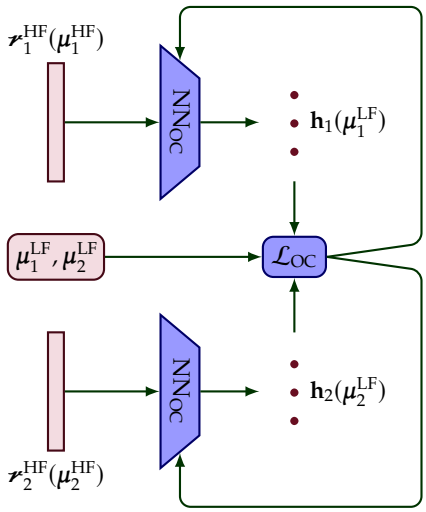


Figure 6.12: Scheme of the Siamese architecture. Red nodes denote the input/output quantities, while blue nodes refer to the relevant computational blocks.

In the present case, $\mathbf{Y}_{\mathrm{OC}}$ comprises $L_{\mathrm{OC}} = 106$ POMs, selected by performing an SVD on $\mathbf{S}_{\mathrm{HF}}$ and choosing $\epsilon_\sigma = 10^{-2}$ as the tolerance in Eq. (3.16). We highlight that the POD tolerance ϵ_σ has been increased compared to that used in deriving $\mathbf{Y}_{\mathrm{LF}}$ due to the different parameterizations underlying $\mathbf{U}^{\mathrm{LF}}$ and $\mathbf{U}^{\mathrm{HF}}$. The damage parametrization results in a higher variance in $\mathbf{S}_{\mathrm{HF}}$, requiring a larger number of POMs to fully capture the structural response under the effect of varying damage and operational conditions. However, $\mathrm{NN}_{\mathrm{OC}}$ aims to capture only the effect of the varying operational conditions.

To encode the operational conditions $\boldsymbol{\mu}^{\mathrm{LF}}$ through $\mathrm{NN}_{\mathrm{OC}}$, we require a distance function $\mathcal{E}_{\mathbf{h}} = \mathcal{E}_{\mathbf{h}}(\mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_1^{\mathrm{HF}}), \mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_2^{\mathrm{HF}}))$ of any pair low-dimensional representations $\mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_1^{\mathrm{HF}}(\boldsymbol{\mu}_1^{\mathrm{HF}}))$ and $\mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_2^{\mathrm{HF}}(\boldsymbol{\mu}_2^{\mathrm{HF}}))$, to approximate the Euclidean distance $\mathcal{E}_{\mathrm{LF}} = \|\boldsymbol{\mu}_1^{\mathrm{LF}} - \boldsymbol{\mu}_2^{\mathrm{LF}}\|_2$. Using the deep metric learning approach proposed in Sec. 5.1, $\mathrm{NN}_{\mathrm{OC}}$ is learned through a *Siamese architecture* (see Fig. 6.12). The required training dataset is generated by augmenting $\mathcal{D}_{\mathrm{HF}}$ through a *pairing process*. For each instance in $\mathcal{D}_{\mathrm{HF}}$, we assemble $N_+ \in \mathbb{N}$ positive pairs and $N_- \in \mathbb{N}$ negative pairs, characterized by similar and dissimilar operational conditions, respectively, as follows:

$$\mathcal{D}_{\mathrm{OC}}^{\mathrm{P}} = \{(\boldsymbol{\varkappa}_1^{\mathrm{HF}}, \boldsymbol{\mu}_1^{\mathrm{LF}}, \boldsymbol{\varkappa}_2^{\mathrm{HF}}, \boldsymbol{\mu}_2^{\mathrm{LF}})_{\tau_{\mathrm{OC}}}\}_{\tau_{\mathrm{OC}}=1}^{N_{\mathrm{OC}}^{\mathrm{P}}}, \tag{6.11}$$

where $N_{\mathrm{OC}}^{\mathrm{P}} = N_{\mathrm{train}}^{\mathrm{HF}}(N_+ + N_-)$ is the total number of pairs. To emulate sensors self-noise, the data in $\mathcal{D}_{\mathrm{HF}}^{\mathrm{P}}$ are then corrupted with an additive Gaussian noise to achieve an SNR of 80.

During training, paired encodings $\boldsymbol{\varkappa}_1^{\mathrm{HF}}$ and $\boldsymbol{\varkappa}_2^{\mathrm{HF}}$ are jointly processed to compute $\mathcal{E}_{\mathbf{h}}(\mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_1^{\mathrm{HF}}), \mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}_2^{\mathrm{HF}}))$ and evalu-

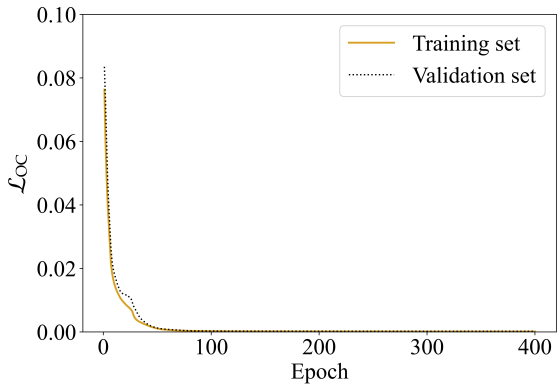


Figure 6.13: L-shaped beam - Learning curve displaying the loss function evolution on the training and validation sets for $\mathrm{NN}_{\mathrm{OC}}$.

ate the following *Square-Square loss function*:

$$\mathcal{L}_{\mathrm{OC}}(\boldsymbol{\Omega}_{\mathrm{OC}}, \mathcal{D}_{\mathrm{OC}}^{\mathrm{P}}) = \frac{1}{N_{\mathrm{OC}}^{\mathrm{P}}} \sum_{\tau_{\mathrm{OC}}=1}^{N_{\mathrm{OC}}^{\mathrm{P}}} \left\{ \frac{1-\Gamma}{2} \mathcal{E}_{\mathbf{h}}^2 + \frac{\Gamma}{2} \left[\max\left(0, \overline{\epsilon_{\mathbf{h}}} - \mathcal{E}_{\mathbf{h}}\right)\right]^2 \right\}_{\tau_{\mathrm{OC}}}, \tag{6.12}$$

which is minimized to tune the learnable weights $\boldsymbol{\Omega}_{\mathrm{OC}}$ that parametrize $\mathrm{NN}_{\mathrm{OC}}$. The computation of $\mathcal{L}_{\mathrm{OC}}$ is carried out by equipping $\mathcal{E}_{\mathbf{h}}$ with the *metric built upon the cosine similarity*, as defined in Eq. (5.5).

The size of the low-dimensional space described by $\mathrm{NN}_{\mathrm{OC}}$ is set to $N_h = 3$. This choice stands in contrast with the findings presented in Sec. 5.3.2, where an optimal size for the low-dimensional space should ideally be consistent with the number of independent parameters within $\boldsymbol{\mu}^{\mathrm{HF}}$ responsible for the data variability — that in this case is 3. Considering the additional normalization constraint associated with the computation of cosine similarity, N_h should be set to 4. However, the choice $N_h = 3$ is motivated by the consideration that the parameters $\boldsymbol{\mu}^{\mathrm{LF}}$ we aim to encode are those associated with highest sensitivity, dominating the feature representation. Therefore, given the parameters of interest $\boldsymbol{\mu}^{\mathrm{LF}}$ and the information loss associated with the linear dimensionality reduction performed via $\mathbf{Y}_{\mathrm{OC}}$, $N_h = 3$ is shown to provide satisfactory feature extraction capabilities, as demonstrated in the following. For the sake of completeness, Fig. 6.13 presents the learning curve of $\mathrm{NN}_{\mathrm{OC}}$ showing the evolution of $\mathcal{L}_{\mathrm{OC}}$ over the training and validation sets. Also in this case, we can appreciate that the model has not encountered overfitting issues. All relevant implementation details and training options for $\mathrm{NN}_{\mathrm{OC}}$ are reported in Appendix B.

For any given set of measurements $\mathbf{U}^{\mathrm{EXP}}$, the trained $\mathrm{NN}_{\mathrm{OC}}$ provides its feature representation $\mathbf{h}^{\mathrm{EXP}}$, as follows:

$$\mathbf{h}^{\mathrm{EXP}} = \mathrm{NN}_{\mathbf{h}}(\mathbf{Y}_{\mathrm{OC}}^{\top} \mathtt{vec}[\mathbf{U}^{\mathrm{EXP}}]). \tag{6.13}$$

$R^2(f_q, \widehat{f_q})$:	0.987	$R^2(Q, \widehat{Q})$:	0.981
$\mathrm{MAE}(f_q, \widehat{f_q})$:	1.28 Hz	$\mathrm{MAE}(Q, \widehat{Q})$:	0.13 kPa

Table 6.4: L-shaped cantilever beam - Accuracy of preliminary identification of the operational conditions.

Given $\mathbf{h}^{\mathrm{EXP}}$, θ_f and θ_Q are identified through a k-NN regressor $\mathcal{M}^{\boldsymbol{\mu}^{\mathrm{LF}}}_{\text{k-NN}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}^{N^{\mathrm{LF}}_{\mathrm{par}}}, \mathbf{h} \mapsto \boldsymbol{\mu}^{\mathrm{LF}}$, applied in the low-dimensional space provided by $\mathrm{NN}_{\mathrm{OC}}$. To calibrate the k-NN algorithm, $\mathcal{D}_{\mathrm{HF}}$ is pre-processed using $\mathrm{NN}_{\mathrm{OC}}$ to derive a new dataset, as follows:

$$\mathcal{D}^{\mathrm{OC}}_{\mathrm{HF}} = \{(\mathbf{h}_{i_{\mathrm{HF}}} = \mathrm{NN}_{\mathrm{OC}}(\boldsymbol{\varkappa}^{\mathrm{HF}}_{i_{\mathrm{HF}}}), \boldsymbol{\mu}^{\mathrm{LF}}_{i_{\mathrm{HF}}})\}^{N^{\mathrm{HF}}_{\mathrm{train}}}_{i_{\mathrm{HF}}=1}, \tag{6.14}$$

which allows fitting $\mathcal{M}^{\boldsymbol{\mu}^{\mathrm{LF}}}_{\text{k-NN}}$ on the data features $\mathbf{h}$ and the corresponding labels $\boldsymbol{\mu}^{\mathrm{LF}}$. The number $N_k \in \mathbb{N}$ of nearest neighbors accounted for in the regression and the associated weighting rule are determined through a *cross-validation strategy*, as described in Sec. 2.2. Considering a 10-fold cross-validation, the maximum fold-averaged regression score, measured via negative RMSE, is achieved with $N_k = 7$ neighbors and inverse distance weighting. Once trained, $\mathcal{M}^{\boldsymbol{\mu}^{\mathrm{LF}}}_{\text{k-NN}}$ estimates the low-fidelity input parameters underlying $\mathbf{U}^{\mathrm{EXP}}$, as $\widehat{\boldsymbol{\mu}}^{\mathrm{LF}} = \mathcal{M}^{\boldsymbol{\mu}^{\mathrm{LF}}}_{\text{k-NN}}(\mathbf{h}^{\mathrm{EXP}})$.

The *low-dimensional embeddings* $\{\mathbf{h}_{i_{\mathrm{HF}}} = (h_1, h_2, h_3)^{\top}_i\}^{N^{\mathrm{HF}}_{\mathrm{train}}}_{i_{\mathrm{HF}}=1}$ provided by $\mathrm{NN}_{\mathrm{OC}}$ for the training data are illustrated in Fig. 6.14a. The two scatter plots include a color channel referring to the target values of the load frequency $\{f_{q\,i_{\mathrm{HF}}}\}^{N^{\mathrm{HF}}_{\mathrm{train}}}_{i_{\mathrm{HF}}=1}$ and of the load amplitude $\{Q_{i_{\mathrm{HF}}}\}^{N^{\mathrm{HF}}_{\mathrm{train}}}_{i_{\mathrm{HF}}=1}$. Note how embeddings associated with different operational conditions feature a relative rotation in the low-dimensional space. This first check ensures that $\mathrm{NN}_{\mathrm{OC}}$ yields a *manifold* to describe the sensitivity of the processed measurements to the varying operational conditions.

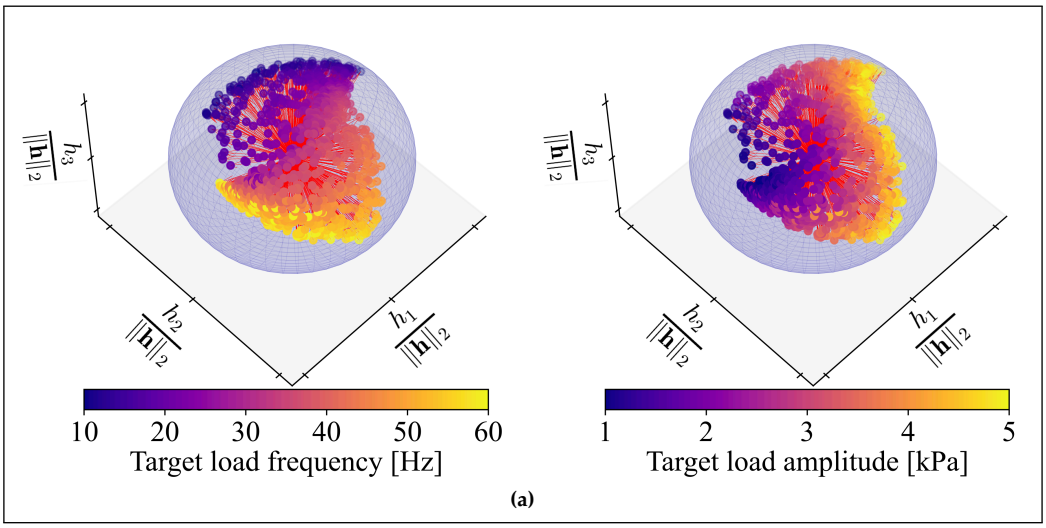
The identification capabilities enabled by $\mathrm{NN}_{\mathrm{OC}}$ and $\mathcal{M}^{\boldsymbol{\mu}^{\mathrm{LF}}}_{\text{k-NN}}$ are assessed on $N^{\mathrm{HF}}_{\mathrm{test}} = 40$ noisy HF data instances. The *parity plots* illustrating the k-NN regression accuracy for the testing data are presented in Fig. 6.14b. The reported values exhibit a very good alignment between predicted and target values, indicating that the operational conditions are mostly identified with high accuracy. This is further confirmed by the corresponding values of *coefficient of determination* R^2* and MAE reported in Tab. 6.4, highlighting a remarkable result given the limited amount of HF data instances used to train $\mathrm{NN}_{\mathrm{OC}}$.

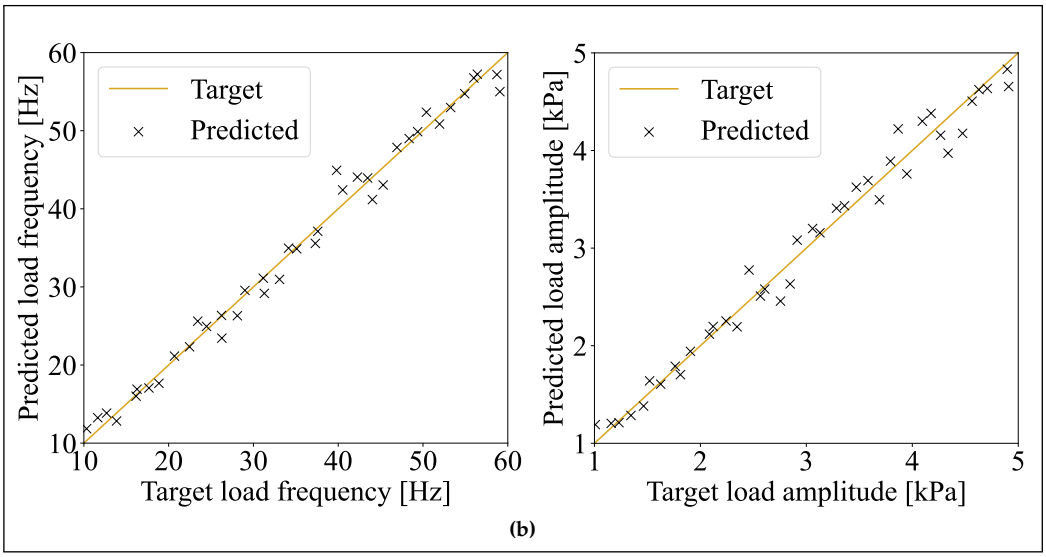
*The coefficient of determination indicates the proportion of variance in the target variable that can be predicted. It is calculated as 1 minus the ratio of the sum of squared prediction errors to the sum of squared differences between the target values and their mean. It can also be expressed as the square of the Pearson correlation coefficient.

6.3.2 Markov Chain Monte Carlo: Simulation Setup

Employing the *Metropolis algorithm* [201] described in Sec. 2.1.2, MCMC updates $p(\boldsymbol{\theta})$ by iteratively generating a chain of $\boldsymbol{\theta}$

201. Metropolis *et al.*, *Equation of State Calculations by Fast Computing Machines*. 1953





samples with length $L_{\text{chain}} \in \mathbb{N}$ from a *proposal distribution*. An *acceptance rule*, similar to that in Eq. (2.13), is employed to determine whether to accept or reject each sample, based on the likelihood of the current sample to represent $\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}$ using NN_{MF}. Assuming an additive Gaussian noise uncorrelated in time to represent internal deviations due to modeling inaccuracies and measurement noise, our *likelihood function* is also Gaussian and reads:

$$p(\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}|\boldsymbol{\theta}) = \prod_{\mathscr{k}=1}^{N_{\text{obs}}} c^{-1}\exp\left(-\frac{\sum_{l=1}^{L}(\mathbf{U}^{\text{EXP}}_{\mathscr{k}} - \widehat{\mathbf{U}}^{\text{HF}})_l^{\top}\boldsymbol{\Sigma}_{\varepsilon}^{-1}(\mathbf{U}^{\text{EXP}}_{\mathscr{k}} - \widehat{\mathbf{U}}^{\text{HF}})_l}{2}\right), \tag{6.15}$$

Figure 6.14: L-shaped cantilever beam - Preliminary identification of the operational conditions: (a) normalized NN_{OC} embeddings of the training data $\{\mathbf{h}_{i_{\text{HF}}} = (h_1, h_2, h_3)_i^{\top}\}_{i_{\text{HF}}=1}^{N^{\text{HF}}_{\text{train}}}$ against the correct values of (left) load frequency $\{f_{q i_{\text{HF}}}\}_{i_{\text{HF}}=1}^{N^{\text{HF}}_{\text{train}}}$ and (right) load amplitude $\{Q_{i_{\text{HF}}}\}_{i_{\text{HF}}=1}^{N^{\text{HF}}_{\text{train}}}$; (b) parity plots of the k-NN regression error on the $N^{\text{HF}}_{\text{test}} = 40$ testing instances for (left) load frequency and (right) load amplitude.

where the denominator $c = (\sqrt{2\pi})^{N_u}\sqrt{|\boldsymbol{\Sigma}_\varepsilon|}$ is a normalization constant. $\boldsymbol{\Sigma}_\varepsilon \in \mathbb{R}^{N_u \times N_u}$ is a covariance matrix derived from an *observation model* that accommodates for *independent measurement errors* and *spatially correlated modeling inaccuracies*, similar to [342]. The (i, j)-th entry of $\boldsymbol{\Sigma}_\varepsilon$ is computed as:

342. Papadimitriou *et al.*, *The effect of prediction error correlation on optimal sensor placement in structural dynamics*. 2012

$$\begin{aligned}\boldsymbol{\Sigma}_\varepsilon(i, j) = \varsigma_{ij} &\left[\mathrm{RMSE}(\mathbf{u}_i^{\mathrm{EXP}}, \widehat{\mathbf{u}}_i^{\mathrm{HF}}) - (1 - \delta_{ij}^{\mathrm{K}})\frac{\mathrm{RMS}(\mathbf{u}_i^{\mathrm{EXP}})}{\sqrt{\mathrm{SNR}}}\right] \\ &\left[\mathrm{RMSE}(\mathbf{u}_j^{\mathrm{EXP}}, \widehat{\mathbf{u}}_j^{\mathrm{HF}}) - (1 - \delta_{ij}^{\mathrm{K}})\frac{\mathrm{RMS}(\mathbf{u}_j^{\mathrm{EXP}})}{\sqrt{\mathrm{SNR}}}\right].\end{aligned} \tag{6.16}$$

In Eq. (6.16), the *root mean square* (RMS) of the prediction error serves as its standard deviation, assuming a zero-mean. The correlation coefficient $\varsigma_{ij} \in \mathbb{R}$ represents the correlation between prediction errors at the i-th and j-th channels, as previously determined by evaluating $\mathrm{NN}_{\mathrm{MF}}$ on the $N_{\mathrm{test}}^{\mathrm{HF}}$ testing instances. The Kronecker delta δ_{ij}^{K} is employed to remove the effect of measurement noise from the off-diagonal elements. Since both $\boldsymbol{\Sigma}_\varepsilon$ and $\boldsymbol{\Sigma}_\varepsilon^{-1}$ depend on $\boldsymbol{\theta}$ via $\widehat{\mathbf{U}}^{\mathrm{HF}}$, they must be computed for each $\boldsymbol{\theta}$ sample. However, this computational overhead does not significantly impact the computational efficiency due to the limited number of channels involved. If necessary, the computation of $\boldsymbol{\Sigma}_\varepsilon^{-1}$ can be speeded up, for instance, by means of an SVD.

Although an informative prior can provide valuable insights into likely damage states, a *uniform prior* is selected for θ_Ω to avoid possible biases in the results. Instead, $\mathrm{NN}_{\mathrm{OC}}$ is used to provide an *informative Gaussian prior* for θ_f and θ_Q. In particular, $p(\theta_f, \theta_Q)$ is centered at the average k-NN regression outcome for the N_{obs} observations, with a covariance matrix derived from the previous (offline) k-NN regression outcomes for the $N_{\mathrm{test}}^{\mathrm{HF}}$ testing instances. Employing this prior to initialize the Markov chains results in a shorter transient phase with fewer samples. Additionally, it constraints the Bayesian inference of $\boldsymbol{\theta}$ within a plausible range, reducing the risk of getting trapped in local maxima of $p(\mathbf{U}_{1,\ldots,N_{\mathrm{obs}}}^{\mathrm{EXP}}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ when making decisions regarding the acceptance of candidate $\boldsymbol{\theta}$ samples.

Fig. 6.15 depicts a sketch of the MCMC procedure. We use the *adaptive Metropolis algorithm* [188], employing a *Gaussian proposal distribution*. This proposal is centered at the previously accepted state and *initialized with diagonal covariance matrix*. The entries of the covariance matrix are initially set small enough so that the sampler gets moving, and are subsequently tuned as the sampling evolves. For the present case, we initialize this covariance matrix by setting the standard deviations for $\{\theta_f, \theta_Q, \theta_\Omega\}$ to 1% of the width of their respective value ranges.

188. Haario *et al.*, *An Adaptive Metropolis Algorithm*. 2001

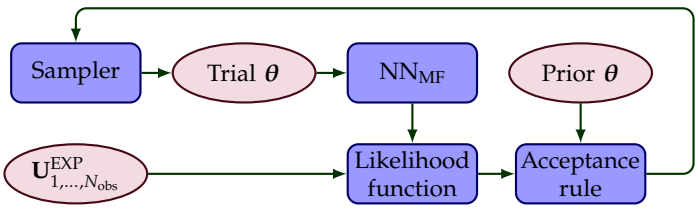


Figure 6.15: Scheme of the MCMC procedure to update the belief about the structural state. Red nodes refer to the input/output quantities, while blue nodes denote the relevant computational blocks.

6.3.3 PARAMETER IDENTIFICATION OUTCOME

The MCMC simulations are carried out using batches of $N_{\mathrm{obs}} = 8$ noisy HF observations. Each observation in the batch corresponds to the same θ_Ω, while θ_f and θ_Q feature a stochastic scattering kept fixed within $(0, T)$. Specifically, each instance in the batch is generated by sampling θ_f and θ_Q from a Gaussian PDF centered at their actual values, with a standard deviation equal to 0.25% of their respective value ranges.

Each MCMC simulation involves generating *five parallel Markov chains*, which are randomly initialized from the prior PDF and simultaneously evolved until they meet the EPSR tolerance set to $R_\varepsilon = 1.01$. The first half of each chain is subsequently discarded to mitigate the *initialization effect*, and 3 out of every 4 samples are further discarded to reduce the *within-chain autocorrelation of samples*.

In the following, we present the outcome of six MCMC analyses carried out under different operational conditions while moving the damage position from the clamp to the free end. Tab. 6.5 reports the results of the damage position identification, in terms of target value, posterior mean, posterior mode, standard deviation, and samples to converge.

In the first three cases, where damage is close to the clamp, its location is accurately identified with relatively low uncertainty. The quality of these estimates is highlighted by the small discrepancy between the target and the posterior mean values, which is approximately 0.2 m. Additionally, it is worth noting the relatively low standard deviation, which, however, increases as the damage moves away from the clamped side. On the other hand, the last three cases, featuring damage farther from the clamped side, yield acceptable estimations but exhibit increased

Case	Target	Mean(θ_Ω)	Mode(θ_Ω)	StdDev(θ_Ω)	L_{chain}
1	0.564 m	0.631 m	0.587 m	0.170 m	2000
2	2.200 m	2.474 m	2.414 m	0.511 m	2000
3	2.888 m	3.088 m	2.844 m	0.710 m	3400
4	4.435 m	4.834 m	4.198 m	0.969 m	2000
5	5.204 m	5.759 m	5.397 m	0.962 m	3000
6	7.380 m	6.080 m	7.136 m	0.866 m	4000

Table 6.5: L-shaped cantilever beam - Damage localization outcomes for MCMC simulations under varying operational and damage conditions.

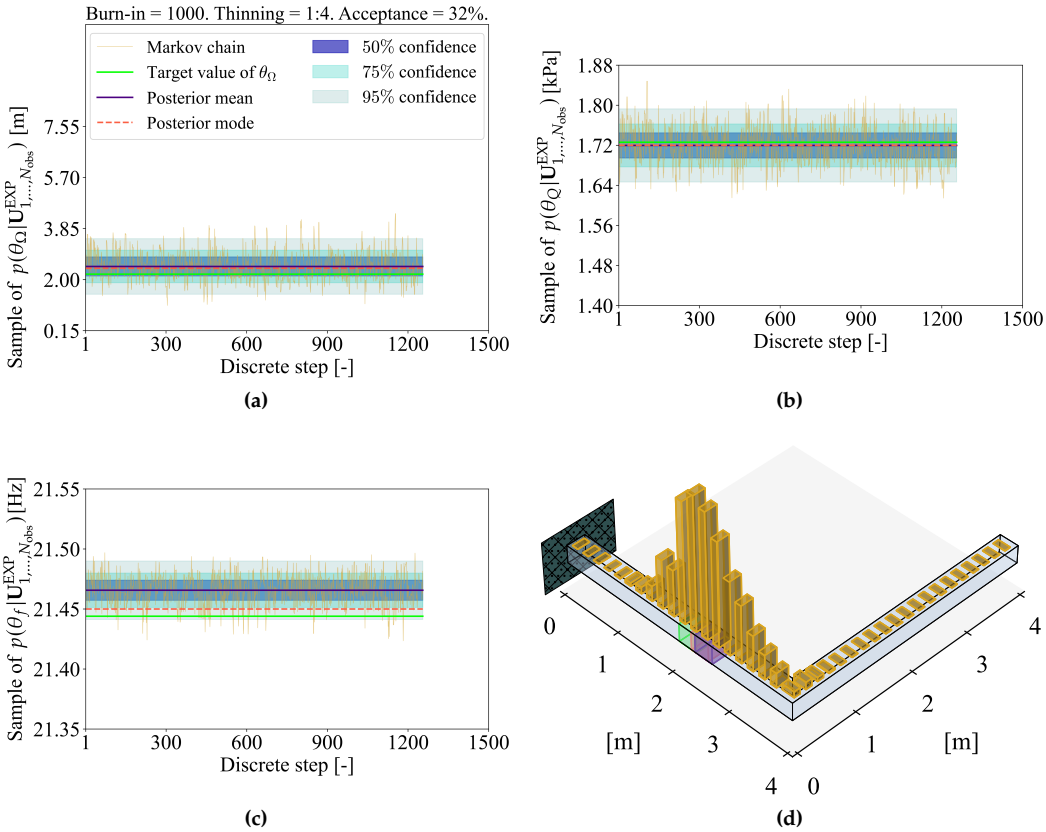


Figure 6.16: L-shaped cantilever beam - Exemplary MCMC outcome (Case 2): Markov chain, target value, posterior mean, posterior mode, and credibility intervals relevant to (a) θ_Ω, (b) θ_Q, and (c) θ_f; (d) histogram of the approximated posterior PDF $p(\theta_\Omega|\mathbf{U}^{\mathrm{EXP}}_{1,\dots,N_{\mathrm{obs}}})$, with details of the target, posterior mean and posterior mode values of θ_Ω, respectively reported as green, purple, and red boxes.

discrepancy between the target and the posterior mean values, along with grater uncertainty. Again, this is due to the limited number of time series containing an imprint of damage. For instance, in Case 5, the discrepancy between the target and the posterior mean values is 0.55 m, while the corresponding standard deviation and credibility intervals increase by 35% compared to those of Case 3. Moreover, we note that Case 6 displays a *skewed distribution*, testified by the increased discrepancy between the posterior mean and mode values, although without a significant increase in standard deviation. In this case, the posterior mode proves to be a more suitable estimate than the posterior mean.

An exemplary MCMC simulation outcome is presented in Fig. 6.16 for Case 2. The graphs display the sampled Markov chain alongside the estimated posterior mean, posterior mode, and credibility intervals for $\{\theta_f, \theta_Q, \theta_\Omega\}$. For θ_f and θ_Q, the chains are plotted over a relatively narrow range of values for visualization purposes. For example, the posterior mean of θ_f exhibits a discrepancy of only 0.02 Hz from the target value, yet falls within the 95% confidence interval. The computing time required for parameter estimation is approximately 130 s,

Case	Target	Mean(θ_Ω)	Mode(θ_Ω)	StdDev(θ_Ω)	L_{chain}
2	2.200 m	2.474 m	2.414 m	0.511 m	2000
2B	2.200 m	2.518 m	2.377 m	0.533 m	2300
2C	2.200 m	2.503 m	2.363 m	0.540 m	12,350

Table 6.6: L-shaped cantilever beam - Impact of the preliminary identification of the operational conditions on the damage localization outcomes.

considering $L_{\text{chain}} = 2000$ samples for 5 *parallel chains*.

Finally, to provide insights into the performance improvement enabled by the preliminary identification of operational conditions, Tab. 6.6 reports additional results for Case 2. Two additional MCMC simulations are carried out, considering uniform priors for all parameters with the Markov chains initialized in high probability regions (Case 2B), and uniform priors for all parameters with the Markov chains randomly initialized (Case 2C), respectively.

We can observe that considering uninformative priors for θ_f and θ_Q, while still using NN_{OC} to initialize the Markov chains in high probability regions, does not significantly affects the results. This is attributed to the large amount of information in each batch of observations, which easily persuades the uninformative priors, as well as the high sensitivity of sensor recordings to the acting load. On the other hand, if we drop the use of NN_{OC} and initialize the Markov chains from uniform distributions, a significantly larger number of samples is required before the initialization effect is overcome and the convergence criterium is met. In this latter case, the computing time required for parameter estimation increases from 130 s to 875 s. Moreover, it has been found necessary to retune the hyperparameters of the covariance matrix of the proposal distribution.

6.4 CONCLUSIONS

In this Chapter, we have proposed a MF setup enabling non-intrusive surrogate modeling for SHM purposes, leveraging model order reduction and artificial neural networks. The MF-DNN-SM maps damage and operational parameters onto sensor recordings as a function of time and parameters. Its multi-level architecture consists of two sequentially trained DNNs: a fully-connected model to replicate sensor recordings in the reference state, and an LSTM model to dynamically enrich the damage-free approximation with the effect of damage. During the supervised learning stage, the effect of damage on vibration response data is accurately replicated through an HF-FE model. Simultaneously, a large LF training dataset, not accounting for damage, is generated using an LF-ROM. By leveraging the correlation across these models, the MF surrogate reduces the

computational cost of offline data generation while ensuring the sensitivity to damage in the approximation. Moreover, it is well-suited for high-dimensional problems and benefits from large low-fidelity training datasets. It adaptively learns both linear and nonlinear correlations, can handle the approximation of strongly discontinuous functions, and provides real-time predictions for new input parameters. Upon detecting a structural deterioration, the digital model can be updated by adjusting only its HF component, with a few new data generated using an updated numerical model — an aspect deemed particularly favorable in view of future developments towards a DT framework, with the goal of devising a SM capable of adapting to the evolving structural health state. The methodology can be adapted to application domains other than SHM, even in the case of a different number of fidelity levels, and potentially be extended to full-field approximations or feature-based data.

The effectiveness and efficiency of the proposed strategy have been assessed on two monitoring scenarios, demonstrating improved approximation capabilities compared to a single-fidelity SM. The computational time required to evaluate the MF surrogate has been reported to be orders of magnitude smaller than that required by the corresponding HF-FE solver. Moreover, we have demonstrated how to integrate the MF surrogate within an MCMC algorithm to sample the posterior PDF of structural health parameters — an exploration useful to validate that the MF-DNN-SM effectively preserves the damage sensitivity of the approximated sensor recordings. Overall, the obtained results are satisfactory in terms of both estimation accuracy and computing time. However, it must be highlighted that this validation has only considered the simpler case of the L-shaped cantilever beam. The considered parameter estimation framework is expected to face limitations in the more complex railway bridge case study, as commented below. This study has also investigated the impact of a preliminary identification of the operational conditions relying on the deep metric learning setting proposed in Chapter 5. This has proved crucial for reducing the computing time required for parameter estimation.

The robustness of the proposed approach to variations in the training datasets or to different damage scenarios is case-specific. As long as the training dataset is sufficiently representative, there are no restrictions on applying the proposed surrogate modeling strategy to large-scale structural systems. However, as the structural or geometrical complexity of the monitored system increases, as observed in the railway bridge case study, it is expected that more (localized) damage patterns may emerge.

To accurately capture and track these patterns, not only in terms of amplitude but also in shape and location, more detailed HF numerical models would be necessary. Consequently, more time-demanding computational models and longer training stages would be required to populate and train on informative enough datasets. This aspect is not only related to the number of instances collected in the training datasets, but also to the number of sensor recordings to be processed and the associated *value of information*. Indeed, damage needs to be sensed via the handled recordings, irrespective of the physical model accuracy, highlighting the importance of *optimal sensor deployment*, observe, e.g., [343,344].

343. Capellari *et al.*, *Structural Health Monitoring Sensor Network Optimization through Bayesian Experimental Design*. 2018

344. Bigoni *et al.*, *Systematic sensor placement for structural anomaly detection in the absence of damaged states*. 2020

While the proposed MF setting is expected to handle even more intricate parametrizations of the structural response, the granted damage identification capabilities might be limited by the employed damage identification setup. This limitation stems from the potentially *slow convergence* of basic MCMC samplers and the *curse of dimensionality*, affecting the inference of low-sensitivity damage parameters. This scenario is likely to occur if the selected damage identification setup was applied to the railway bridge case study. To address this challenge, improvements can be achieved through the use of more advanced samplers, such as the transitional MCMC [177] or No-U-Turn [192] algorithms, or by exploiting data features to describe the structural response.

177. Betz *et al.*, *Transitional Markov Chain Monte Carlo: Observations and Improvements*. 2016

192. Hoffman *et al.*, *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*. 2014

The considerations above motivate the developments that we present in the next Chapter. There, the proposed surrogate modeling strategy is exploited to generate extremely large labeled datasets instead of directly addressing the parameter estimation problem. These data are then exploited to train two *learnable mappings* relying on DL models. Specifically, a feature extractor and a feature-oriented SM are trained on data features to be learned from the sensor recordings, and then synergistically leveraged within the MCMC sampler. Exploiting learnable features in place of raw vibration recordings is shown to provide significant improvements in the parameter identification accuracy, as well as considerable gains in efficiency due to the low-dimensionality of the involved features.

This page intentionally left blank.

7 ENHANCING BAYESIAN MODEL UPDATING VIA DEEP NETS

Part of the research presented in this Chapter has been published as

[166] *Torzoni et al., Enhancing Bayesian model updating in structural health monitoring via learnable mapping, 2024.* Under review.

In this Chapter, we present a *hybrid* data-driven/model-based, supervised, stochastic strategy for real-time damage localization and quantification. We leverage the strengths of both data-driven and model-based approaches to inverse problems by integrating an *MCMC framework* with the *automatic feature extraction* capabilities enabled by DL models. Automatic feature extraction allows for features optimized for the task at hand. Simultaneously, Bayesian model-based inference of SHM parameters naturally addresses the ill-posed nature of the SHM problem, allowing for the quantification of the associated uncertainties. The methodology is highly flexible, enabling easy adaptation to handle data of varying nature or to solve inverse problems in application domains beyond SHM, such as scattering problems, medical diagnoses, and inverse kinematics.

A graphical abstraction of the proposed framework is depicted in Fig. 7.1. Within a *simulation-based SHM framework*, during an *initial offline phase*, we employ physics-based full and reduced-order models to derive an MF-DNN-SM, following the strategy proposed in Chapter 6. The MF-DNN-SM is employed to efficiently generate a large labeled dataset for training two data-driven models based on DNNs: a *feature extractor* and a *feature-oriented SM*. Before feature extraction, sensor recordings are pre-processed via *Markov transition field* [160] to transform vibration recordings into images. Representing time series as images enhances the visibility of *local patterns* in the data that might otherwise be obscured or spread over time. The feature extractor is devised to extract data features by mapping input MTF images to their *feature representation in a low-dimensional space*. It employs an AE architecture for *dimensionality reduction* of the input data, equipped with a *Siamese appendix of the encoder* optimized through *pairwise contrastive learning*. Similar to the *deep metric learning framework* proposed in Chapter 5, the feature extractor encodes the sensitivity to the sought parameters in the learned metric space, enabling a *manifold* to capture the parametric space describing the variability of the structural system. On the other hand, the feature oriented SM is designed

160. Wang *et al., Imaging Time-Series to Improve Classification and Imputation.* 2015

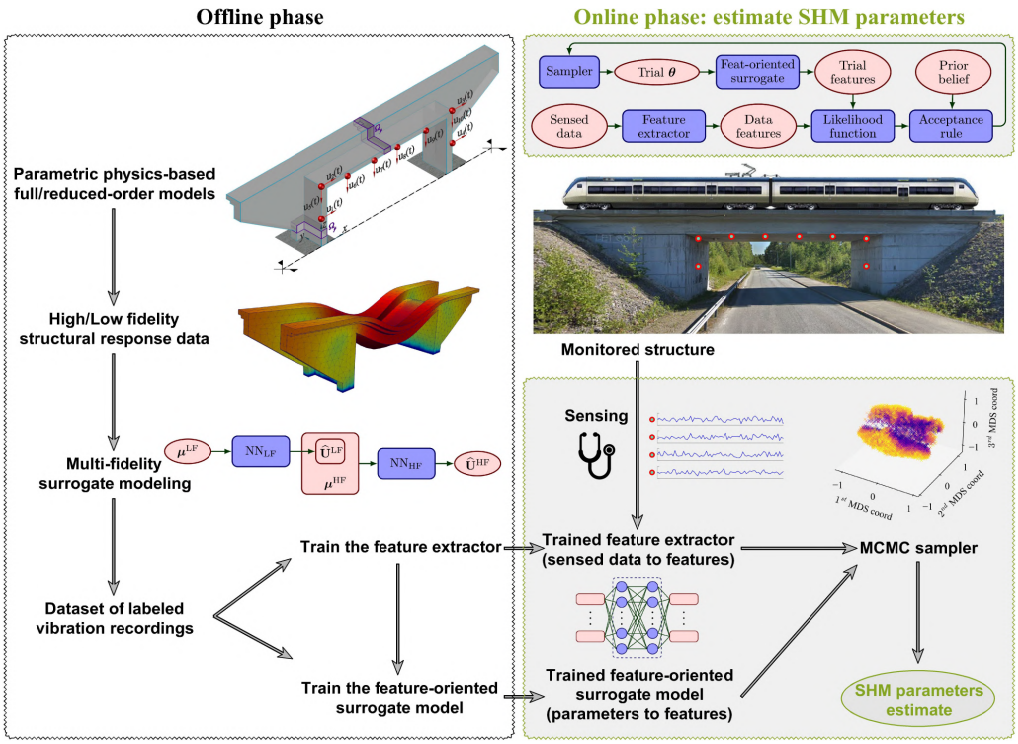


Figure 7.1: Enhanced Bayesian model updating: graphical abstraction.

to approximate the functional link between the parameters to be updated and the low-dimensional feature space described by the feature extractor. During the *online monitoring phase*, these two data-driven models work in tandem to evaluate a likelihood function underlying an MCMC sampler and eventually update the prior belief about the structural state.

The novelty of the proposed methodology lies in assimilating vibration response data through the combined use of DL-based feature extraction and MCMC sampling. The low-dimensional feature space incorporates a *geometric structure* reflecting the sensitivity to the parameters to be updated. This enables an *enhanced MCMC framework* that enjoys competitive computational costs due to the low dimensionality of the involved features, fast convergence thanks to the geometric structure characterizing the feature space, and accurate estimates resulting from the informativeness of the extracted features.

The parameter estimation capabilities of the proposed approach are assessed through the simulated monitoring of three structures: a cantilever beam, a portal frame (not considered in the previous two Chapters), and a railway bridge. The obtained results demonstrate that employing learnable features instead of raw vibration recordings significantly improves the parameter identification performance achieved in Sec. 6.3.3.

The Chapter is organized as follows. In Sec. 7.1, we describe the proposed methodology to enhance Bayesian model updating through *learnable mappings*. We detail the composition of the training data generated with the MF-DNN-SM. We then focus on the architectures and technical aspects related to the training and evaluation of the feature extractor and of the feature-oriented SM in Sec. 7.1.1 and Sec. 7.1.2, respectively. We discuss how these two models are employed to enhance the MCMC-based parameter estimation framework in Sec. 7.1.3. In Sec. 7.2, we assess the computational procedure on three case studies of increasing structural complexity. Finally, Sec. 7.3 discusses the obtained results and draws the conclusions.

7.1 FEATURE-BASED MARKOV CHAIN MONTE CARLO SAMPLING

We consider an SHM setup that relies on the assimilation of vibration recordings $\mathbf{U}^{\text{EXP}}(\boldsymbol{\theta}) = [\mathbf{u}_1^{\text{EXP}}(\boldsymbol{\theta}), \dots, \mathbf{u}_{N_u}^{\text{EXP}}(\boldsymbol{\theta})] \in \mathbb{R}^{L \times N_u}$, consisting of N_u series of L measurements equally spaced in time. Each recording refers to a time interval $(0, T)$, under assumed steady operational and damage conditions. The vector $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$ consists of $N_\theta \in \mathbb{N}$ parameters, representing the variability of the monitored system in terms of structural health and, possibly, operational conditions, for which we seek to update the relative belief. The damage region is parametrized through its position $\mathbf{y}$ and magnitude δ, similar to Chapter 5 and Chapter 6.

Training data associated with specific damage conditions are generated in an inexpensive way using the MF-DNN surrogate modeling strategy proposed in Chapter 6. In particular, by taking $\boldsymbol{\mu}^{\text{HF}}$, with $\boldsymbol{\theta} \subseteq \boldsymbol{\mu}^{\text{HF}}$, as uniformly distributed and sampling it via Latin hypercube rule, the trained MF-DNN-SM is exploited to populate a dataset collecting N_{train} training instances as:

$$\mathcal{D}_{\text{MF}} = \{(\widehat{\mathbf{U}}_i^{\text{HF}} = \text{NN}_{\text{MF}}(\boldsymbol{\mu}_i^{\text{HF}}), \boldsymbol{\mu}_i^{\text{HF}})\}_{i=1}^{N_{\text{train}}}. \tag{7.1}$$

To mimic measurement noise, the vibration recordings in Eq. (7.1) are then corrupted by adding an independent, identically distributed Gaussian noise.

One of the key aspects of NN_{MF} is that it is trained on a small dataset of HF structural responses, where the impact of damage is replicated with an HF-FE model considered to provide an accurate description of potential damage scenarios. Additionally, as discussed in Sec. 6.2, the computational time to evaluate NN_{MF} for new input parameters is negligible. This capability enables to greatly speed up the generation of a large dataset compared to relying solely on the HF-FE model.

The recent advancements in computer vision suggest the possibility of transforming time series onto images for SHM purposes, see, e.g., [307,308,309]. *Imaging time series* enables to highlight local patterns that might otherwise be spread over or lie outside the time domain. In this context, we employ the *MTF technique* [160] to convert the vibration recordings in $\mathcal{D}_{\mathrm{MF}}$ into images before training the feature extractor and the feature-oriented SM. MTF is preferred over other conversion methods, such as *Gramian angular fields* [160], *recurrence plots* [345] and *grey-scale encoding* [346], due to its reported superior performance for SHM purposes [308, 309]. However, it is worth noting that MTF is not as commonly used as signal processing methods based on spectral analysis, such as *spectrogram* or *scalogram* representations.

307. Giglioni *et al.*, *Deep Autoencoders for Unsupervised Damage Detection with Application to the Z24 Benchmark Bridge*. 2023

308. Garcia *et al.*, *Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms*. 2022

309. Mantawy *et al.*, *Convolutional neural network based structural health monitoring for rocking bridge system by encoding time-series into images*. 2022

160. Wang *et al.*, *Imaging Time-Series to Improve Classification and Imputation*. 2015

345. Marwan *et al.*, *Recurrence plots for the analysis of complex systems*. 2007

346. Xu *et al.*, *Online Fault Diagnosis Method Based on Transfer Convolutional Neural Networks*. 2020

The MTF technique is reviewed in Appendix C. In essence, it first discretizes the value range of time series data into a set of symbolic states. It then computes the Markov transition matrix, measuring the probabilities of moving from one state to another. Finally, it spreads these probabilities over the temporal axis to measure the state transition probability for each pair of time steps, not necessarily consecutive. We stress that this choice is not restrictive, and the proposed methodology can be readily adapted to accommodate data with different characteristics.

Each instance $\widehat{\mathbf{U}}_i^{\mathrm{HF}}$, with $i = 1, \ldots, N_{\mathrm{train}}$, is transformed into a *grey-scale mosaic* $\mathcal{I}_i \in \mathbb{R}^{N_R^{\mathcal{I}} \times N_C^{\mathcal{I}}}$, where $N_R^{\mathcal{I}} \in \mathbb{N}$ and $N_C^{\mathcal{I}} \in \mathbb{N}$ respectively denote the height and the width of the mosaic. Each mosaic is the juxtaposition of N_u MTF representations, or *tesserae*, obtained via MTF encoding of the N_u time series in $\widehat{\mathbf{U}}_i^{\mathrm{HF}}$. Thus, $\mathcal{D}_{\mathrm{MF}}$ is reassembled as:

$$\mathcal{D}_{\mathcal{I}} = \{(\mathcal{I}_i, \boldsymbol{\mu}_i^{\mathrm{HF}})\}_{i=1}^{N_{\mathrm{train}}}. \tag{7.2}$$

It is worth noting that an alternative viable approach might be to stack the N_u MTF representations in the out-of-plane direction instead of composing a mosaic by juxtaposing them. Both cases can be effectively processed using *2D convolutional layers*.

In the following two Sections, we describe the models and the training process underlying the feature extractor and the feature-oriented surrogate. The two models are learned through a *sequential training process* involving two learning steps. The *first step* involves training the feature extractor to distill informative features from sensed data. The *second step* involves training the surrogate model to map the $\boldsymbol{\theta}$ parameters onto the low-dimensional feature space. Once trained, the two models serve to sample the posterior distribution of $\boldsymbol{\theta}$ conditioned on observational data $\mathbf{U}^{\mathrm{EXP}}$ through an MCMC algorithm.

7.1.1 FEATURE EXTRACTOR MODEL

The feature extractor is built upon a *2D convolutional AE*, equipped with a *Siamese appendix* [156] of the encoder branch (refer to "Training 1" in Fig. 7.2). This model enhances the dimensionality reduction capabilities provided by the *unsupervised training* of AEs, discussed in Sec. 4.2.7, by incorporating a *distance function* for the latent space through *pairwise contrastive learning* [154], as proposed in Sec. 5.1. In the latent space, features from *similar data* are pushed to be as close as possible, while those from *dissimilar data* are pushed apart. Here, the notion of similarity refers to a *task-specific distance measure* in terms of the $\boldsymbol{\theta}$ parameters.

156. Bromley *et al.*, *Signature Verification using a "Siamese" Time Delay Neural Network*. 1993

154. Hadsell *et al.*, *Dimensionality Reduction by Learning an Invariant Mapping*. 2006

The learnable components of the feature extractor are the *encoder* $\mathrm{NN}_{\mathrm{ENC}}$ and *decoder* $\mathrm{NN}_{\mathrm{DEC}}$ branches of an AE $\mathrm{NN}_{\mathrm{CAE}}$ — with the subscript CAE standing for "contrastive autoencoder". The encoder $\mathrm{NN}_{\mathrm{ENC}} : \mathbb{R}^{N_R^{\mathcal{I}} \times N_C^{\mathcal{I}}} \mapsto \mathbb{R}^{N_h}$, $\mathcal{I} \mapsto \mathbf{h}$, provides the feature representation $\mathbf{h}$ of the input mosaic $\mathcal{I}$ in a low-dimensional space of size $N_h \in \mathbb{N}$. On the other hand, the decoder $\mathrm{NN}_{\mathrm{DEC}} : \mathbb{R}^{N_h} \mapsto \mathbb{R}^{N_R^{\mathcal{I}} \times N_C^{\mathcal{I}}}$, $\mathbf{h} \mapsto \mathcal{I}$, takes $\mathbf{h}$ to reconstruct the input mosaic $\mathcal{I}$. The *bottleneck layer* linking $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{DEC}}$ is characterized by the *low-dimensional feature size* N_h, which compresses the data while attempting to replicate the input as closely as possible at the output.

The training of $\mathrm{NN}_{\mathrm{ENC}}$ involves minimizing a standard *reconstruction loss*, similar to that in Eq. (4.27). Additionally, the Siamese appendix of the encoder affects the training process through a *contrastive loss term* that connects two twins $\mathrm{NN}_{\mathrm{ENC}}$, requiring a *data pairing process* to augment $\mathcal{D}_{\mathcal{I}}$. Accordingly, a *threshold distance* $\overline{\mathcal{E}_{\boldsymbol{\theta}}}$ is fixed to characterize the similarity of data points in terms of $\boldsymbol{\theta}$. Subsequently, for each mosaic in $\mathcal{D}_{\mathcal{I}}$, we assemble N_+ *positive pairs* characterized by $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 \leq \overline{\mathcal{E}_{\boldsymbol{\theta}}}$, and N_- *negative pairs* characterized by $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 > \overline{\mathcal{E}_{\boldsymbol{\theta}}}$, as follows:

$$\mathcal{D}_{\mathcal{I}}^{\mathrm{P}} = \{(\mathcal{I}_1, \boldsymbol{\mu}_1^{\mathrm{HF}}, \mathcal{I}_2, \boldsymbol{\mu}_2^{\mathrm{HF}})_\tau\}_{\tau=1}^{N_{\mathrm{train}}^{\mathrm{P}}}, \tag{7.3}$$

where $N_{\mathrm{train}}^{\mathrm{P}} = N_{\mathrm{train}}(N_+ + N_-)$ denotes the *total number of pairs*.

During "Training 1", *paired mosaics* $\mathcal{I}_1(\boldsymbol{\mu}_1^{\mathrm{HF}})$ and $\mathcal{I}_2(\boldsymbol{\mu}_2^{\mathrm{HF}})$ are jointly processed to compute the associated distance function $\mathcal{E}_{\mathbf{h}}(\mathrm{NN}_{\mathrm{ENC}}(\mathcal{I}_1), \mathrm{NN}_{\mathrm{ENC}}(\mathcal{I}_2))$. The weights $\boldsymbol{\Omega}_{\mathrm{CAE}}$ parametrizing $\mathrm{NN}_{\mathrm{CAE}}$ are then optimized minimizing the following loss:

$$\begin{aligned}
&\mathcal{L}_{\mathrm{CAE}}(\boldsymbol{\Omega}_{\mathrm{CAE}}, \mathcal{D}_{\mathcal{I}}^{\mathrm{P}}) = \\
&\frac{1}{N_{\mathrm{train}}^{\mathrm{P}}} \sum_{\tau=1}^{N_{\mathrm{train}}^{\mathrm{P}}} \Bigg\{ \|\mathrm{vec}[\mathcal{I}_1(\boldsymbol{\mu}_1^{\mathrm{HF}}) - \mathrm{NN}_{\mathrm{DEC}}(\mathrm{NN}_{\mathrm{ENC}}(\mathcal{I}_1(\boldsymbol{\mu}_1^{\mathrm{HF}})))]\|_2^2 + \\
&\qquad \left[\frac{1-\Gamma}{2}\mathcal{E}_{\mathbf{h}}^2 + \frac{\Gamma}{2}\left[\max\left(0, \overline{\epsilon_{\mathbf{h}}} - \mathcal{E}_{\mathbf{h}}\right)\right]^2\right] \Bigg\}_\tau .
\end{aligned} \tag{7.4}$$

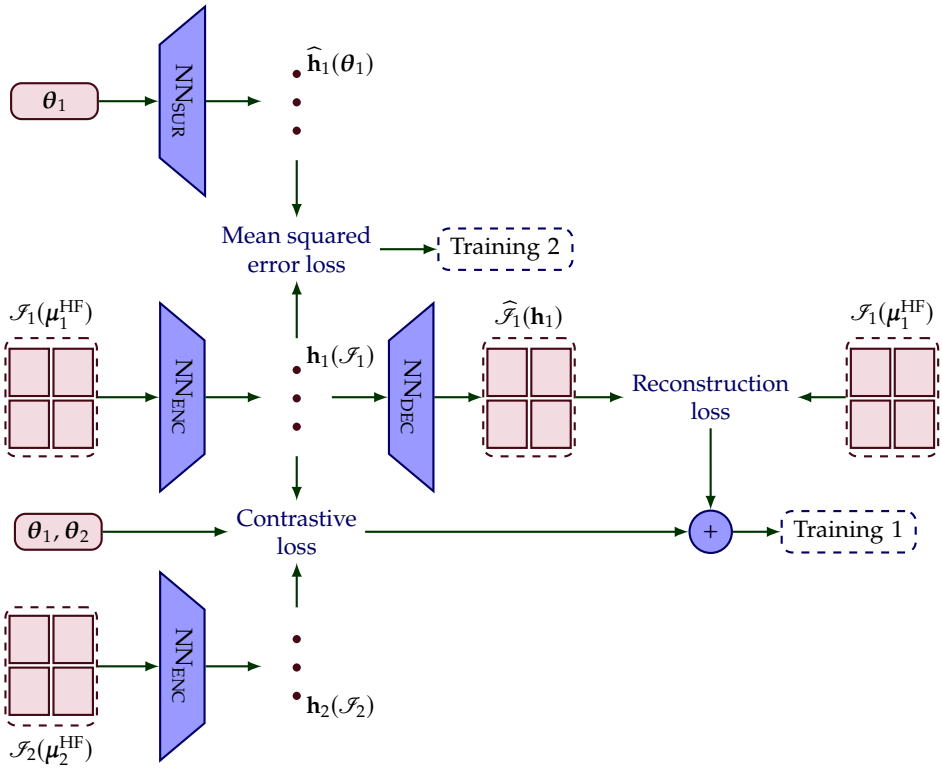


Figure 7.2: Learnable feature extractor and feature-oriented surrogate: flowchart of the sequential training process. Red nodes refer to the input/output quantities, while blue nodes denote the relevant computational blocks. Hat variables denote quantities obtained from neural network approximations. $\mathrm{NN}_{\mathrm{ENC}}$ denotes the feature extractor, $\mathrm{NN}_{\mathrm{DEC}}$ denotes the decoder branch, and $\mathrm{NN}_{\mathrm{SUR}}$ denotes the feature-oriented surrogate model. $\mathscr{I}(\boldsymbol{\mu}^{\mathrm{HF}})$ denotes an input mosaic, $\widehat{\mathscr{I}}(\mathbf{h})$ denotes a reconstructed mosaic, and $\boldsymbol{\theta}$ denotes the vector of parameters for which we seek to update the relative belief. $\mathbf{h}(\mathscr{I})$ is the low-dimensional feature representation of $\mathscr{I}(\boldsymbol{\mu}^{\mathrm{HF}})$ provided by $\mathrm{NN}_{\mathrm{ENC}}$, and $\widehat{\mathbf{h}}(\boldsymbol{\theta})$ is the corresponding approximation provided by $\mathrm{NN}_{\mathrm{SUR}}$.

In Eq. (7.4), the first term is an *MSE-based reconstruction loss function*, similar to that Eq. (4.27), while the second term is the *pairwise Square-Square loss function*, useful to induce a geometric structure in the feature space. Minimizing $\mathcal{L}_{\mathrm{CAE}}$ ensures that the feature representation provided by $\mathrm{NN}_{\mathrm{ENC}}$ enables recovering the input via $\mathrm{NN}_{\mathrm{DEC}}$ while concentrating on a manifold that captures the sensitivity to $\boldsymbol{\theta}$. The label information is exploited to guide the dimensionality reduction, resulting in the distance function $\mathcal{E}_{\mathbf{h}}$ approximating the Euclidean distance $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2$. The contrastive term is computed by equipping $\mathcal{E}_{\mathbf{h}}$ with the *Euclidean metric*, as defined in Eq. (5.5).

For any given input mosaic $\mathscr{I}(\boldsymbol{\mu}^{\mathrm{HF}})$, the trained $\mathrm{NN}_{\mathrm{CAE}}$ can be evaluated as a standard AE to provide its feature representation $\mathbf{h}(\mathscr{I})$ and the corresponding reconstruction $\widehat{\mathscr{I}}(\mathbf{h})$, as follows:

$$\mathbf{h}(\mathscr{I}) = \mathrm{NN}_{\mathrm{ENC}}(\mathscr{I}(\boldsymbol{\mu}^{\mathrm{HF}})), \tag{7.5}$$

$$\widehat{\mathscr{I}}(\mathbf{h}) = \mathrm{NN}_{\mathrm{DEC}}(\mathbf{h}(\mathscr{I})). \tag{7.6}$$

7.1.2 Feature-oriented surrogate model

After the first learning step, the Siamese appendix as well as $\mathcal{D}_{\mathscr{I}}^{\mathrm{P}}$ are discarded, retaining only $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathcal{D}_{\mathscr{I}}$ to train the feature-oriented surrogate $\mathrm{NN}_{\mathrm{SUR}}$ (refer to "Training 2" in Fig. 7.2).

$\mathrm{NN}_{\mathrm{SUR}} : \mathbb{R}^{N_\theta} \mapsto \mathbb{R}^{N_h}$, $\boldsymbol{\theta} \mapsto \mathbf{h}$, is a *fully-connected DL model* designed to approximate the functional relationship between the parametric space of $\boldsymbol{\theta}$ and the low-dimensional feature space described by $\mathrm{NN}_{\mathrm{ENC}}$. The required training dataset is obtained by pre-processing $\mathcal{D}_{\mathcal{I}}$ through $\mathrm{NN}_{\mathrm{ENC}}$. In particular, by mapping the mosaics in $\mathcal{D}_{\mathcal{I}}$ onto the learned feature space, a new dataset dedicated to train $\mathrm{NN}_{\mathrm{SUR}}$ is obtained as:

$$\mathcal{D}_{\boldsymbol{\theta}}^{\mathbf{h}} = \{(\boldsymbol{\theta}_i, \mathbf{h}_i = \mathrm{NN}_{\mathrm{ENC}}(\mathcal{I}_i))\}_{i=1}^{N_{\mathrm{train}}}, \tag{7.7}$$

collecting the parameters $\boldsymbol{\theta}$ for which we aim to update the relative belief and the corresponding labels, in terms of the feature representations $\mathbf{h}$ of the training data.

The set of weights $\boldsymbol{\Omega}_{\mathrm{SUR}}$ that parameterizes $\mathrm{NN}_{\mathrm{SUR}}$ is learned through the minimization of the following loss function:

$$\mathcal{L}_{\mathrm{SUR}}(\boldsymbol{\Omega}_{\mathrm{SUR}}, \mathcal{D}_{\boldsymbol{\theta}}^{\mathbf{h}}) = \frac{1}{N_{\mathrm{train}}} \sum_{i=1}^{N_{\mathrm{train}}} \|\mathbf{h}_i(\mathcal{I}_i) - \mathrm{NN}_{\mathrm{SUR}}(\boldsymbol{\theta}_i)\|_2^2, \tag{7.8}$$

providing a measure of the distance between the target low-dimensional features vector $\mathbf{h}(\mathcal{I})$, obtained through $\mathrm{NN}_{\mathrm{ENC}}$, and its approximated counterpart, provided by $\mathrm{NN}_{\mathrm{SUR}}$.

For any given vector of parameters $\boldsymbol{\theta}$, the trained $\mathrm{NN}_{\mathrm{SUR}}$ provides the feature representation of a (possible) corresponding mosaic, as follows:

$$\widehat{\mathbf{h}} = \mathrm{NN}_{\mathrm{SUR}}(\boldsymbol{\theta}), \tag{7.9}$$

where $\widehat{\mathbf{h}}$ denotes the $\mathrm{NN}_{\mathrm{SUR}}$ approximation to the low-dimensional features provided by $\mathrm{NN}_{\mathrm{ENC}}$.

The algorithmic description of the procedures and computations characterizing the preliminary offline phase of the proposed strategy is reported in Algorithm 7.1. After training, $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$ are jointly exploited in an online monitoring phase to enhance an MCMC sampler for model updating purposes, as described in the next Section.

It is worth noting that the modeling choices for $\mathrm{NN}_{\mathrm{CAE}}$ and $\mathrm{NN}_{\mathrm{SUR}}$ are tailored to the specific characteristics of the considered observational data. However, the overall framework is quite general, allowing for easy adaptation to different modeling choices that suit the characteristics of the specific problem at hand. In our case, vibration data undergo MTF pre-processing to highlight structures and patterns hidden within the data. While we thus show how to extract informative features in a low-dimensional metric space in the case of image data, data of various natures can be similarly addressed by selecting appropriate DL architectures. Additionally, there may be cases where the decoding branch $\mathrm{NN}_{\mathrm{DEC}}$ should be discarded. The

Algorithm 7.1 Enhanced Bayesian model update via deep nets. Preliminary offline phase: algorithmic description

Input: parametrization of the physics-based models
trained MF-DNN-SM $\mathrm{NN}_{\mathrm{MF}}$
training options for the contrastive AE $\mathrm{NN}_{\mathrm{CAE}}$
training options for the feature-oriented SM $\mathrm{NN}_{\mathrm{SUR}}$

1: use $\mathrm{NN}_{\mathrm{MF}}$ to populate the training dataset $\mathcal{D}_{\mathrm{MF}}$
2: use the MTF encoding to derive $\mathcal{D}_{\mathcal{J}}$ from $\mathcal{D}_{\mathrm{MF}}$
3: augment $\mathcal{D}_{\mathcal{J}}$ to $\mathcal{D}^{\mathrm{P}}_{\mathcal{J}}$ through the pairing process
4: train $\mathrm{NN}_{\mathrm{CAE}}$ ("Training 1") on $\mathcal{D}^{\mathrm{P}}_{\mathcal{J}}$
5: use the trained $\mathrm{NN}_{\mathrm{ENC}}$ to derive $\mathcal{D}^{\mathbf{h}}_{\boldsymbol{\theta}}$ from $\mathcal{D}_{\mathcal{J}}$
6: train $\mathrm{NN}_{\mathrm{SUR}}$ ("Training 2") on $\mathcal{D}^{\mathbf{h}}_{\boldsymbol{\theta}}$
7: **return** trained feature extractor $\mathrm{NN}_{\mathrm{ENC}}$,
8: trained feature-oriented SM $\mathrm{NN}_{\mathrm{SUR}}$

reason for retaining $\mathrm{NN}_{\mathrm{DEC}}$ lies in the fact that the unsupervised reconstruction term in Eq. (4.27) serves to regularize the overall learning process. Furthermore, the trained $\mathrm{NN}_{\mathrm{DEC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$ can be used to establish a surrogate model of the type $\mathrm{NN}_{\mathrm{DEC}}(\mathrm{NN}_{\mathrm{SUR}}(\boldsymbol{\theta}))$, similar to the approach proposed in [134]. In our case, the contrastive term in Eq. (7.4) is minimized exploiting label information that fully describes the parametrization underlying the physics-based modeling of the problem. However, when the number $N^{\mathrm{HF}}_{\mathrm{par}}$ of parameters in $\boldsymbol{\mu}^{\mathrm{HF}}$ becomes large, the paring process involved in minimizing $\mathcal{L}_{\mathrm{CAE}}$ becomes computationally demanding. Although this is not our case, this issue could be addressed by including only a subset of $\boldsymbol{\mu}^{\mathrm{HF}}$ in $\boldsymbol{\theta}$, limited to the parameters for which we seek to update the relative belief. In this eventuality, the decoding branch $\mathrm{NN}_{\mathrm{DEC}}$ should be omitted to avoid a latent space exhibiting a dependence on parameters not included in $\boldsymbol{\theta}$, and preventing a one-to-one correspondence between $\boldsymbol{\theta}$ and $\mathbf{h}$, which may not be captured by $\mathrm{NN}_{\mathrm{SUR}}$. The same consideration applies to systems subject to unknown stochastic inputs, such as seismic or wind loads acting on civil structures, where the dependency of $\mathbf{h}$ on parameters describing stochastic inputs can not be uniquely defined and might not be modeled accurately by $\mathrm{NN}_{\mathrm{SUR}}$.

134. Fresca *et al.*, *Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Nonlinear Time-Dependent Parametrized PDEs.* 2021

7.1.3 Markov Chain Monte Carlo: Simulation Setup

Once trained, $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$ serve to enhance an MCMC sampler for model updating purposes. The updating procedure follows the setup described in Sec. 2.1.2 and already employed in Sec. 6.3. Specifically, we use the *adaptive Metropolis algorithm* to update the prior PDF $p(\boldsymbol{\theta})$ and provide a posterior $p(\boldsymbol{\theta}|\mathbf{U}^{\mathrm{EXP}}_{1,\dots,N_{\mathrm{obs}}})$, conditioned on a batch of $N_{\mathrm{obs}} \in \mathbb{N}$ mea-

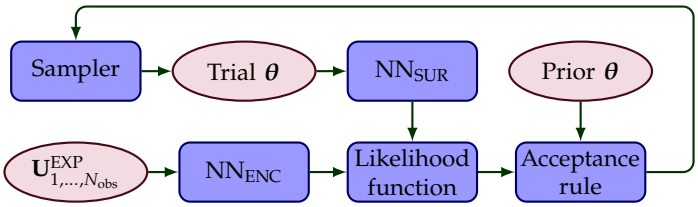


Figure 7.3: Enhanced Bayesian Model Updating - Scheme of the MCMC procedure to update the belief about the structural state. Red nodes refer to the input/output quantities, while blue nodes denote computational blocks.

surements $\mathbf{U}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}} = [\mathbf{U}^{\mathrm{EXP}}_1, \ldots, \mathbf{U}^{\mathrm{EXP}}_{N_{\mathrm{obs}}}] \in \mathbb{R}^{N_{\mathrm{obs}} \times L \times N_u}$, under assumed steady damage conditions. We remark that the proposed procedure is general, allowing for different choices of the sampling algorithm. While more effective samplers exist in the literature, such as [189,190,191,192], we choose to rely on a simple yet flexible algorithm and enhance Bayesian model updating by exploiting a *data-driven perspective*.

189. Geman *et al.*, *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*. 1984

190. Ching *et al.*, *Transitional Markov Chain Monte Carlo Method for Bayesian Model Updating, Model Class Selection, and Model Averaging*. 2007

191. Green *et al.*, *Delayed Rejection in Reversible Jump Metropolis-Hastings*. 2001

192. Hoffman *et al.*, *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*. 2014

The updating procedure is carried out based on the *likelihood* of the current $\boldsymbol{\theta}$ sample to represent $\mathbf{U}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$. To this aim, $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$ are synergistically exploited to provide informative features $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$ by assimilating $\mathbf{U}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$ and to surrogate the functional link between $\boldsymbol{\theta}$ and $\mathbf{h}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}$, respectively, as illustrated in Fig. 7.3. The resulting parameter estimation framework enjoys a *competitive computational cost* due to the small size N_h of the feature space described by $\mathrm{NN}_{\mathrm{ENC}}$, as well as the negligible cost required to evaluate $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$. It also benefits from an *improved convergence rate* thanks to the contrastive learning of $\mathrm{NN}_{\mathrm{ENC}}$, and *accurate estimates* due to the informativeness of the extracted features.

By assuming an additive Gaussian noise to represent the uncertainty arising from modeling inaccuracies and measurement noise, the *likelihood function* also becomes Gaussian and reads:

$$p(\mathbf{U}^{\mathrm{EXP}}_{1,\ldots,N_{\mathrm{obs}}}|\boldsymbol{\theta}) = \prod_{\mathscr{k}=1}^{N_{\mathrm{obs}}} c^{-1} \exp\left(- \frac{(\mathbf{h}^{\mathrm{EXP}}_{\mathscr{k}} - \widehat{\mathbf{h}}(\boldsymbol{\theta}))^{\top}(\mathbf{h}^{\mathrm{EXP}}_{\mathscr{k}} - \widehat{\mathbf{h}}(\boldsymbol{\theta}))}{2\sigma_{\varepsilon}^2} \right), \tag{7.10}$$

where the denominator $c = \sqrt{2\pi\sigma_{\varepsilon}^2}$ is as a normalization constant, and $\sigma_{\varepsilon} = \mathrm{RMSE}(\mathbf{h}^{\mathrm{EXP}}, \widehat{\mathbf{h}}(\boldsymbol{\theta}))$ serves as the standard deviation of the uncertainty under the *zero-mean assumption*. As for the likelihood function in Eq. (6.15), due to its dependence on $\boldsymbol{\theta}$, σ_{ε} must be computed for each $\boldsymbol{\theta}$ sample. However, this does not significantly impact the overall computational cost due to the low dimensionality of the feature space.

The *proposal PDF* is taken as Gaussian, centered at the previously accepted state and *initialized with diagonal covariance matrix*. Initially, the entries of the covariance matrix are set small enough so that the sampler gets moving, and then tuned as the sampling evolves. Each MCMC simulation involves gen-

erating *five parallel Markov chains*, which are randomly initialized from the prior PDF and simultaneously evolved to satisfy the EPSR tolerance set to $R_\varepsilon = 1.01$. The first half of each chain is subsequently discarded to mitigate the *initialization effect*, and 3 out of every 4 samples are further discarded to reduce the *within-chain autocorrelation*.

7.2 PARAMETER IDENTIFICATION: NUMERICAL EXPERIMENTS

The software and hardware setup used to implement and evaluate the FOMs and ROMS underlying Problem (3.1) and Problem (3.18), as well as the architectures of NN_{CAE} and NN_{SUR}, are the same as those exploited in the previous Chapters. The implementation details of NN_{CAE} and NN_{SUR} are provided in Appendix D. For all experiments, the size of the low-dimensional space described by NN_{ENC} is set to $N_h = 20$.

7.2.1 L-SHAPED CANTILEVER BEAM

The first test case we consider is the L-shaped beam, previously addressed in Sec. 5.3 and Sec. 6.2.1. The considered SHM setup is identical to that employed in Sec. 6.2.1, see also Fig. 5.2, and we thus avoid to re-describe it.

The MF-DNN surrogate model used to generate $\mathcal{D}_{\text{MF}}$ is the same NN_{MF} model described in Sec. 6.2.1. In essence, NN_{MF} approximates displacement time histories for $N_u = 8$ DOFs, mimicking the monitoring system depicted in Fig. 5.2. Recordings are provided for a time interval $(0, T = 1\text{ s})$ with an acquisition frequency of $f_s = 200$ Hz. The vector of HF input parameters is $\boldsymbol{\mu}^{\text{HF}} = (Q, f_q, x_\Omega, y_\Omega)^\top$, where $Q \in [1, 5]$ kPa and $f_q \in [10, 60]$ Hz represent the load amplitude and frequency, respectively, and $\mathbf{y} = (x_\Omega, y_\Omega)^\top$ encodes the position of a damage region $\Omega_{\mathbf{y}}$ where the material stiffness is reduced by $\delta = 25\%$, with either x_Ω or y_Ω varying in the range $[0.15, 3.85]$ m. By taking $\boldsymbol{\mu}^{\text{HF}}$ as uniformly distributed and sampling it via Latin hypercube rule, $\mathcal{D}_{\text{MF}}$ is populated with $N_{\text{train}} = 20,000$ instances. Recordings are then corrupted with a Gaussian noise to achieve an SNR of 100.

Before training NN_{CAE} and NN_{SUR}, the vibration recordings in $\mathcal{D}_{\text{MF}}$ are encoded into images via MTF to derive $\mathcal{D}_{\mathcal{I}}$. Each Mosaic $\mathcal{I}_i$ in $\mathcal{D}_{\mathcal{I}}$, with $i = 1, \dots, N_{\text{train}}$, is obtained by arranging the $N_u = 8$ MTF tesserae into a 2×4 grid, with each MTF tessera being a 40×40 pixel image, as shown in Fig. 7.4 for an example. The size of the MTF tesserae depends on the length L of the time series and the width of the blurring kernel. For the

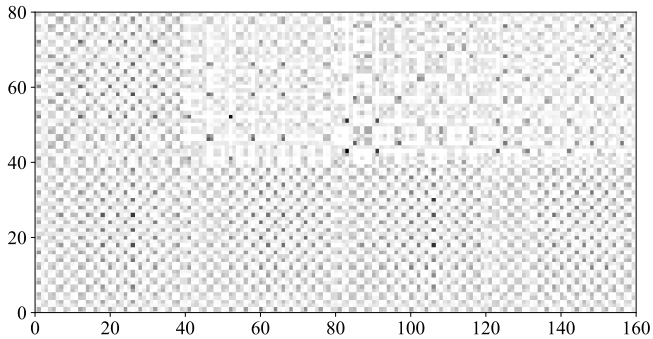


Figure 7.4: L-shaped cantilever beam - Exemplary MTF mosaic.

detailed steps behind the MTF encoding, refer to Appendix C. In this case, the length L of the vibration recordings in $\mathcal{D}_{\text{MF}}$ is reduced by removing the initial 20% of each time history to eliminate potential inaccuracies induced by the LSTM model underlying NN_{HF}. The chosen width of the blurring kernel is equal to 4. Additionally, each recording in $\mathcal{D}_{\text{MF}}$ is normalized to follow a standard Gaussian distribution, thus allowing us to neglect the dependence on the load amplitude Q due to the linear-elastic modeling behind $\mathcal{D}_{\text{HF}}$. The mosaics dataset $\mathcal{D}_{\mathcal{J}}$ is then employed to minimize the loss functions in Eq. (7.4) and Eq. (7.8), as described in Sec. 7.1.1 and Sec. 7.1.2, respectively.

A compact representation of the low-dimensional features extracted through NN_{ENC} for the validation set of $\mathcal{D}_{\mathcal{J}}$ is depicted in Fig. 7.5. These scatter plots report a *downsized MDS representation* of the extracted features $\{\mathbf{h}_i\}_{i=1}^{N_{\text{val}}}$, with $N_{\text{val}} \in \mathbb{N}$ being the number of validation data points. In these plots, the color channel corresponds to the target values of the load frequency $\{f_{qi}\}_{i=1}^{N_{\text{val}}}$ and of the damage position along the x and y directions $\{x_{\Omega i}\}_{i=1}^{N_{\text{val}}}$ and $\{y_{\Omega i}\}_{i=1}^{N_{\text{val}}}$, respectively. Note how the resulting manifold effectively encodes the sensitivity of the processed measurements on the parameters employed to describe the variability of the system. This visual check offers an initial qualitative indication of the positive impact of adopting the feature space described by NN_{ENC} for addressing the foreseen Bayesian model updating task, as it features a smooth and regularly organized structure.

The parameters for which we aim to update the relative belief are given as $\boldsymbol{\theta} = (\theta_f, \theta_\Omega)^\top$, where the target position of $\Omega_{\mathbf{y}}$ is encoded by the abscissa $\theta_\Omega \in [0.15, 7.55]$ m. The MCMC simulations are carried out using batches of $N_{\text{obs}} = 8$ HF noisy observations. Each observation in the batch is relative to the same $\Omega_{\mathbf{y}}$, while Q and f_q are sampled from a Gaussian PDF centered at their ground truth values, with a standard deviation equal to 0.25% of their respective vale ranges.

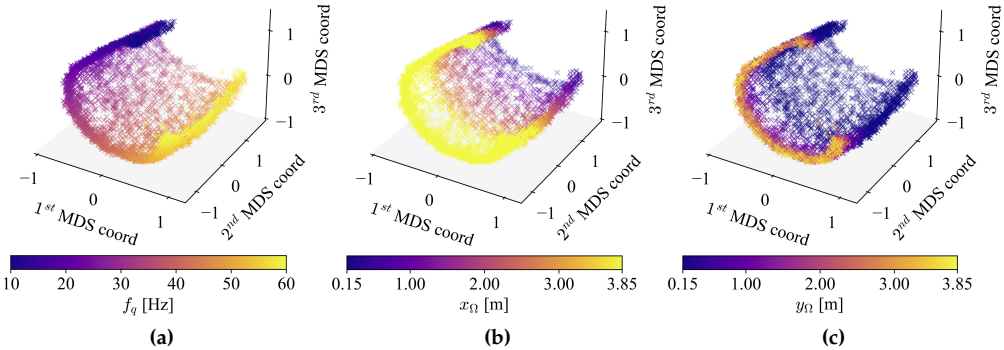


Figure 7.5: L-shaped cantilever beam - Multidimensional scaling onto three dimensions of the NN_{ENC} embeddings for the validation set $\{\mathbf{h}_i\}_{i=1}^{N_{\text{val}}}$, against the target values of: (a) load frequency $\{f_{qi}\}_{i=1}^{N_{\text{val}}}$; (b) damage position along the x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{val}}}$; (c) damage position along the y-direction $\{y_{\Omega i}\}_{i=1}^{N_{\text{val}}}$.

In the following, we present the outcome of six MCMC simulations carried out under different operational conditions while moving the damage position from the clamp to the free-end. Tab. 7.1 presents the outcomes of the damage position identification, in terms of target value, posterior mean, posterior mode, standard deviation, and samples to converge.

The quality of the estimates is highlighted by the small discrepancy between the target and the posterior mean values, which is only a few centimeters (less than 3% of the admissible support length). Additionally, note the relatively low values of standard deviation, which however increase as the damage position moves farther from the clamped side due to a reduced sensitivity of sensor recordings to damage.

The only situation characterized by a significant difference between the target and the posterior mean values, as well as by a larger uncertainty, is in fact Case 6, which features a damage position close to the free-end of the beam. For instance, in Case 4, the discrepancy between the target and the posterior mean values is only 0.044 m over an allowable support length of 7.4 m, while it reaches 0.845 m in Case 6. Despite the larger discrepancy between the target and the posterior mean, the target value falls within the 95% confidence interval, as in the other cases, demonstrating the reliability of the estimates provided.

The obtained results offer the opportunity for a direct comparison with the former MCMC outcomes presented in Sec. 6.3.3 and reported in Tab. 6.5. In the latter, $p(\boldsymbol{\theta}|\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$ was sampled directly using NN_{MF} instead of employing NN_{ENC} and NN_{SUR}. This comparison reveals that NN_{ENC} and NN_{SUR} enable improving the parameter identification outcomes in all the considered performance indicators, as well as significantly reducing the number of samples required to meet the EPSR criterium. In Case 4, for instance, the discrepancy between the target and the posterior mean values, and the standard deviation value, are shown in Tab. 7.1 to decrease by 7 and 10 times, respectively.

Case	Target	Mean(θ_Ω)	Mode(θ_Ω)	StdDev(θ_Ω)	L_{chain}
1	0.564 m	0.580 m	0.600 m	0.043 m	2200
2	2.200 m	2.195 m	2.225 m	0.110 m	2000
3	2.888 m	2.830 m	2.887 m	0.137 m	2000
4	4.435 m	4.391 m	4.362 m	0.077 m	2000
5	5.204 m	5.403 m	5.412 m	0.315 m	2150
6	7.380 m	6.535 m	6.200 m	0.511 m	2250

Table 7.1: L-shaped cantilever beam - Damage localization outcomes for enhanced MCMC simulations under varying operational and damage conditions.

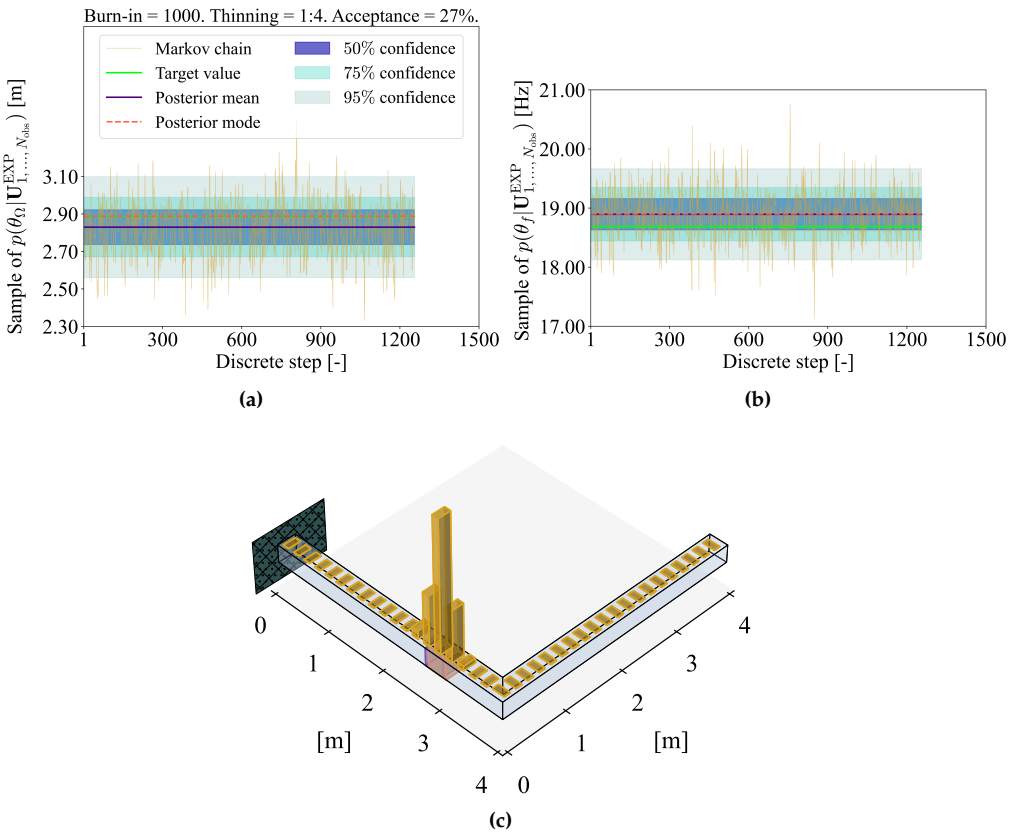


Figure 7.6: L-shaped cantilever beam - Exemplary enhanced MCMC outcome (Case 3): Markov chain, target value, posterior mean, posterior mode, and credibility intervals relevant to (a) damage position θ_Ω and (b) load frequency θ_f; (c) histogram of the approximated posterior PDF $p(\theta_\Omega|\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$, with details of the target, posterior mean and posterior mode values of θ_Ω, respectively reported as green, purple, and red boxes.

An exemplary MCMC simulation outcome is presented in Fig. 7.6 for Case 3. The graphs display the sampled Markov chain alongside the estimated posterior mean, posterior mode, and credibility intervals, for both θ_Ω and θ_f. Note that the chains are plotted over a relatively small range of values for the sake of visualization. Thanks to the low-dimensionality of the features, the procedure also enjoys a considerable computational efficiency. The computing time required for parameter estimation is only approximately 5 s. Compared to the MCMC outcomes presented in Sec. 6.3.3, the achieved speed-ups range from 25 to 50 times approximately. This is a remarkable result, highlighting the real-time damage identification capabilities of the proposed strategy, all with quantified uncertainty.

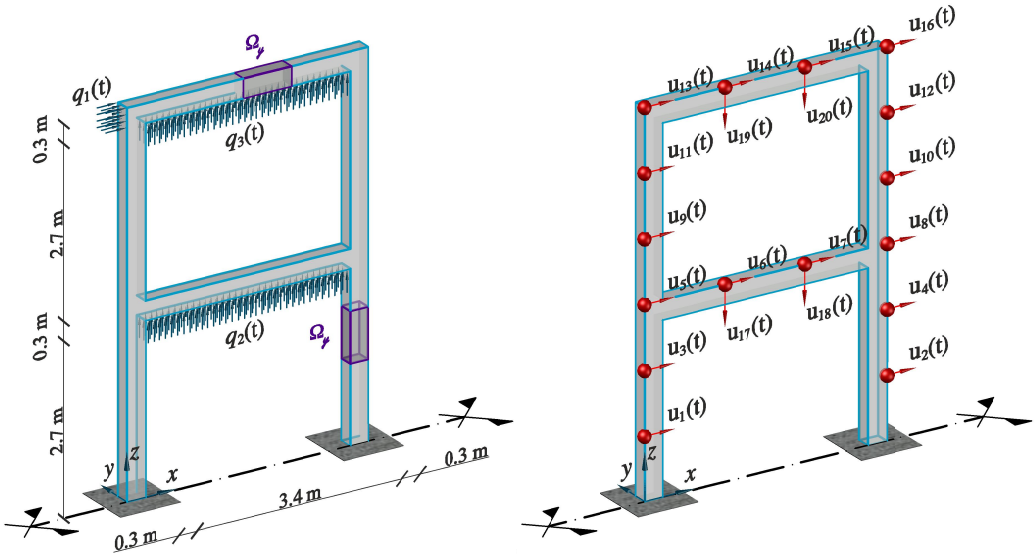


Figure 7.7: Portal frame: details of (a) loading condition, damageable region Ω_y, and (b) synthetic recordings related to displacements $u_1(t), \ldots, u_{20}(t)$.

7.2.2 PORTAL FRAME

The second test case involves the two-story portal frame depicted in Fig. 7.7. The columns have a width of 0.3 m, the beams have a height of 0.3 m, the inter-story height is 2.7 m, the span of the beams is 3.4 m, and the out of plane thickness is 0.45 m. The assumed mechanical properties are: Young's modulus $E = 34$ GPa, Poisson's ratio $\nu = 0.2$, density $\rho = 2500$ kg/m^3. The structure is excited by three distributed loads $q_1(t)$, $q_2(t)$, and $q_3(t)$, respectively applied on top of the left column and on the bottom face of the two horizontal beams, as shown in Fig. 7.7a. The three loads vary in time according to:

$$q_{1,2,3}(t) = \begin{cases} Q\frac{t}{T_q}, & \text{if } t \leq T_q, \\ 0, & \text{if } t > T_q, \end{cases} \tag{7.11}$$

with $Q = 10$ kPa and $T_q = 0.08$ s. This *linear-ramp actuation* may be connected to smart instrumented structures, equipped with an excitation system designed for *forced vibration tests*.

Displacement time histories are obtained from $N_u = 20$ DOFs, mimicking a monitoring system deployed as depicted in Fig. 7.7b. Each recording spans a time interval $(0, T = 1.12$ s) with a sampling frequency of $f_s = 125$ Hz, and it is corrupted with an additive Gaussian noise, yielding an SNR of 150.

The HF-FE model is established through a FE discretization using linear tetrahedral elements, resulting in $N_{\text{FE}} = 4827$ DOFs. The Rayleigh's damping matrix is assembled to account for a 2.5% damping ratio on the first two structural modes. Damage

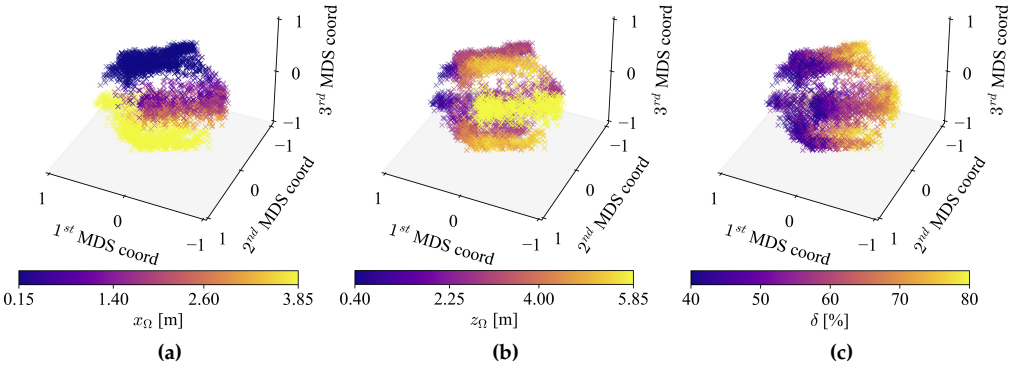


Figure 7.8: Portal frame - Multidimensional scaling onto three dimensions of the $\mathrm{NN_{ENC}}$ embeddings for the validation set $\{\mathbf{h}_i\}_{i=1}^{N_{\mathrm{val}}}$, against the target values of: (a) damage position along the x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\mathrm{val}}}$; (b) damage position along the z-direction $\{z_{\Omega i}\}_{i=1}^{N_{\mathrm{val}}}$; (c) damage level $\{\delta_i\}_{i=1}^{N_{\mathrm{val}}}$.

is simulated by means of a localized stiffness reduction that can occur anywhere in the frame, within subdomains $\Omega_{\mathbf{y}}$ featuring a different layout for the columns and for the beams (see Fig. 7.7a). The position of $\Omega_{\mathbf{y}}$ is parameterized through $\mathbf{y} = (x_\Omega, z_\Omega)^\top$, with either x_Ω or z_Ω varying in the ranges $[0.15, 3.85]$ m and $[0.4, 5.85]$ m, respectively. In this case, the magnitude of the stiffness reduction can range in $\delta \in [40\%, 80\%]$, and it remains constant during an excitation event.

According to the MF setting proposed in Sec. 6.1, the LF dataset $\mathcal{D}_{\mathrm{LF}}$ consists of a single instance representing the structural response in the absence of damage, as the LF structural response is not parametrized. This instance is directly employed to replace $\mathrm{NN_{LF}}$ in $\mathrm{NN_{MF}}$. On the other hand, the HF dataset $\mathcal{D}_{\mathrm{HF}}$ is populated with $N_{\mathrm{train}}^{\mathrm{HF}} = 1000$ HF-FE data instances for training $\mathrm{NN_{HF}}$ to incorporate the effects of damage into the LF structural response for any given $\boldsymbol{\mu}^{\mathrm{HF}} = (x_\Omega, z_\Omega, \delta)^\top$. Once trained, $\mathrm{NN_{MF}}$ is then employed to populate $\mathcal{D}_{\mathrm{MF}}$ with $N_{\mathrm{train}} = 20,000$ instances for varying values of $\boldsymbol{\mu}^{\mathrm{HF}}$.

Dataset $\mathcal{D}_{\mathcal{J}}$ is obtained by encoding each instance in $\mathcal{D}_{\mathrm{MF}}$ into a 4×5 MTF mosaic, where each MTF tessera is a 32×32 pixel image. Similar to the previous case, before undergoing MTF encoding, the vibration recordings in $\mathcal{D}_{\mathrm{MF}}$ are normalized to follow a Gaussian distribution with zero mean and unit standard deviation. Additionally, the initial 8% of each time history is removed to eliminate potential inaccuracies induced by $\mathrm{NN_{HF}}$. In this case, the width of the blurring kernel is set to 4.

The MDS plots of the features $\{\mathbf{h}_i\}_{i=1}^{N_{\mathrm{val}}}$ provided by $\mathrm{NN_{ENC}}$ for the validation set of $\mathcal{D}_{\mathcal{J}}$ are depicted in Fig. 7.8. Here, the color channels correspond to the damage position along the x and z directions $\{x_{\Omega i}\}_{i=1}^{N_{\mathrm{val}}}$ and $\{z_{\Omega i}\}_{i=1}^{N_{\mathrm{val}}}$, respectively, and to the damage level $\{\delta_i\}_{i=1}^{N_{\mathrm{val}}}$. Once again, these plots qualitatively show the presence of an underlying manifold that encodes the sensitivity of the recordings to the structural health parameters.

Case	Target($\theta_x;\theta_z;\theta_\delta$)	Mean($\theta_x;\theta_z;\theta_\delta$)	Mode($\theta_x;\theta_z;\theta_\delta$)	StdDev($\theta_x;\theta_z;\theta_\delta$)	L_{chain}
1	0.15 m; 0.58 m; 74.32%	0.21 m; 0.66 m; 74.54%	0.15 m; 0.40 m; 76.00%	0.06 m; 0.22 m; 2.21%	4550
2	0.15 m; 3.68 m; 77.69%	0.21 m; 3.47 m; 74.92%	0.15 m; 3.67 m; 76.00%	0.06 m; 0.16 m; 3.29%	3350
3	3.85 m; 2.65 m; 67.58%	3.76 m; 2.61 m; 68.63%	3.85 m; 2.58 m; 68.00%	0.07 m; 0.19 m; 2.65%	4050
4	3.85 m; 4.94 m; 53.30%	3.76 m; 5.04 m; 53.13%	3.85 m; 5.30 m; 52.00%	0.07 m; 0.19 m; 4.02%	5850
5	1.94 m; 2.85 m; 56.64%	1.85 m; 2.84 m; 56.37%	1.63 m; 3.58 m; 56.00%	0.26 m; 0.23 m; 2.99%	3850
6	1.70 m; 5.85 m; 63.70%	1.90 m; 5.70 m; 69.06%	2.00 m; 5.85 m; 68.00%	0.29 m; 0.16 m; 3.35%	2950

Table 7.2: Portal frame - Damage localization and quantification outcomes for enhanced MCMC simulations under varying damage conditions.

The feature space described by NN_{ENC} is employed together with NN_{SUR} to update the prior belief of $\boldsymbol{\theta} = (\theta_x, \theta_z, \theta_\delta)^\top$ via MCMC sampling. In contrast to the L-shaped beam case, encoding the target position of $\Omega_{\mathbf{y}}$ using a single abscissa is not suitable here, requiring two parameters to describe it. Additionally, the magnitude of the applied stiffness reduction δ is not fixed, but it is treated as an uncertain parameter of interest. The MCMC algorithm is fed with batches of $N_{\text{obs}} = 8$ HF noisy observations, all related to the same damage location and magnitude.

The results obtained from six MCMC simulations are summarized in Tab. 7.2. Overall, both the damage location and magnitude are identified with very high accuracy and relatively low uncertainty. There are no cases characterized by a significant discrepancy between the target and the posterior mean values. As expected, the standard deviation of either θ_x or θ_z is larger along the axis in which $\Omega_{\mathbf{y}}$ can move. Additionally, the uncertainty in θ_δ increases as $\Omega_{\mathbf{y}}$ gets far from the clamped sides, due to the smaller sensitivity of sensor recordings to damage in such cases. For visualization purposes, we report an exemplary MCMC-recovered posterior distribution in Fig. 7.9 for Case 3. Note that the chains for θ_x and θ_z are plotted over a relatively small range of values for sake of graphical clarity.

7.2.3 RAILWAY BRIDGE

This third case study involves the railway bridge previously considered in Sec. 5.4 and Sec. 6.2.2. The considered SHM setup is identical to that employed in Sec. 6.2.2, as illustrated in Fig. 6.7.

The dataset $\mathcal{D}_{\text{MF}}$ is generated using the same NN_{MF} model described in Sec. 6.2.2. In summary, NN_{MF} approximates displacement time histories for $N_u = 10$ DOFs. Recordings are provided for a time interval $(0, T = 1.5\text{ s})$ with an acquisition frequency of $f_s = 400$ Hz. The vector of HF input parameters is $\boldsymbol{\mu}^{\text{HF}} = (v, m_A, x_\Omega, z_\Omega, \delta)^\top$, where $v \in [160, 215]$ km/h and $m_A \in [16, 22]$ ton represent the train velocity and the axle-carried mass, respectively; and $\mathbf{y} = (x_\Omega, z_\Omega)^\top$ encodes the position of a damage region $\Omega_{\mathbf{y}}$ where the material stiffness reduction can occur with a magnitude $\delta \in [40\%, 80\%]$, with either x_Ω or z_Ω varying in the range $[-0.115, 16.515]$ m and $[0.4, 6.25]$ m, respec-

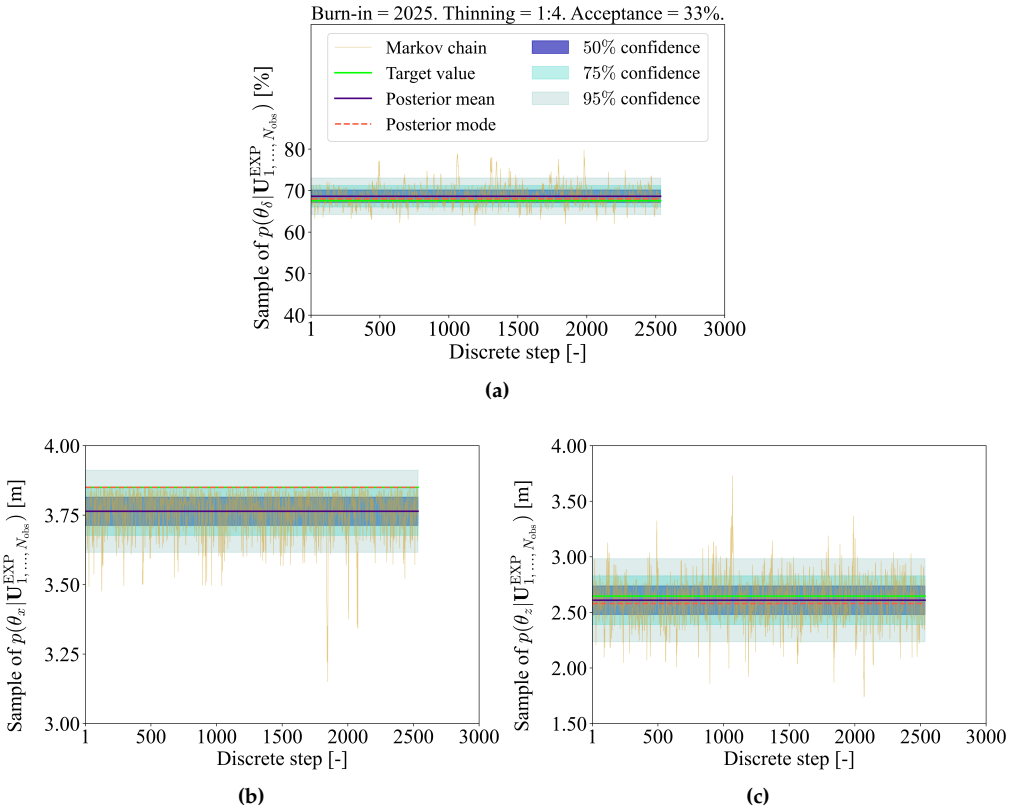


Figure 7.9: Portal frame - Exemplary enhanced MCMC outcome (Case 3): Markov chain, target value, posterior mean, posterior mode, and credibility intervals relevant to (a) damage level θ_δ and (b) damage position along the x-direction θ_x; (c) damage position along the z-direction θ_z.

tively. Compared to what presented in Sec. 6.2.2, here $\mathrm{NN}_{\mathrm{MF}}$ is trained considering an HF-FE model that features a 5% damping ratio on the first two structural modes instead of the first four, and the range of values in which δ can vary is increased from $\delta \in [20\%, 50\%]$ to $\delta \in [40\%, 80\%]$. Once trained, $\mathrm{NN}_{\mathrm{MF}}$ is employed to populate $\mathcal{D}_{\mathrm{MF}}$ with $N_{\mathrm{train}} = 30,000$ training instances for varying values of $\boldsymbol{\mu}^{\mathrm{HF}}$.

The mosaics dataset $\mathcal{D}_{\mathcal{J}}$ is obtained by encoding each training instance in $\mathcal{D}_{\mathrm{MF}}$ into a 2×5 MTF mosaic, with each MTF tessera being a 64×64 pixel image. Before undergoing MTF encoding, the initial 4% of each time history in $\mathcal{D}_{\mathrm{MF}}$ is removed to eliminate potential inaccuracies induced by $\mathrm{NN}_{\mathrm{HF}}$. Moreover, since in this case the vibration recordings are characterized by data distributions mainly spread over the tails, as shown by the HF signals in Fig. 6.8, each time history in $\mathcal{D}_{\mathrm{MF}}$ is normalized to range between 0 and 1, and quantized through a uniform bin assignment instead of a Gaussian one. In this case, the chosen width of the blurring kernel is equal to 9.

The visual check on the MDS representations of the features provided by $\mathrm{NN}_{\mathrm{ENC}}$ for the validation set is depicted in Fig. 7.10. Here, the different color channels refer to each entry of $\boldsymbol{\mu}^{\mathrm{HF}}$.

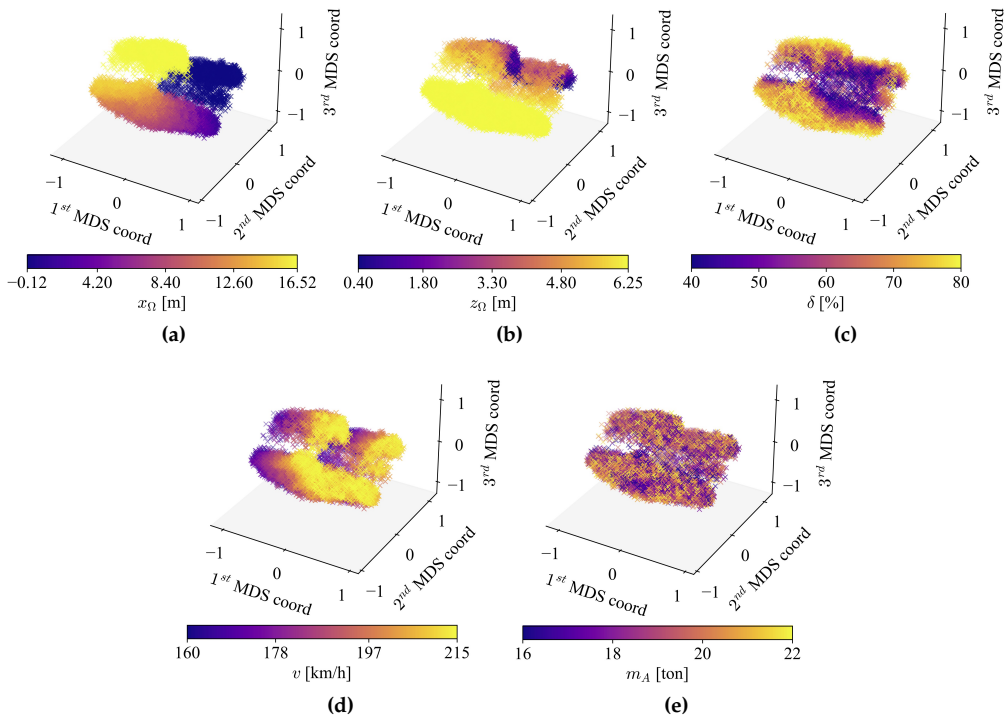


Figure 7.10: Railway bridge - Multidimensional scaling onto three dimensions of the NN_{ENC} embeddings for the validation set $\{\mathbf{h}_i\}_{i=1}^{N_{\text{val}}}$, against the target values of: (a) damage position along the x-direction $\{x_{\Omega i}\}_{i=1}^{N_{\text{val}}}$; (b) damage position along the z-direction $\{z_{\Omega i}\}_{i=1}^{N_{\text{val}}}$; (c) damage level $\{\delta_i\}_{i=1}^{N_{\text{val}}}$; (d) train velocity $\{v_i\}_{i=1}^{N_{\text{val}}}$; (e) axle-carried mass $\{m_{Ai}\}_{i=1}^{N_{\text{val}}}$.

It is interesting to note how the overall shape defined by the scatter plots resembles the structural layout of the bridge (rotated and extruded), which is automatically retrieved from x_Ω and z_Ω. These plots qualitatively demonstrate a clear sensitivity of the low-dimensional feature space to the damage location, the damage magnitude, and the train velocity. However, the axle-carried mass is characterized by a fuzzier representation, lacking a manifold topology capable of adequately capturing its influence on the processed measurements.

The parameters for which we seek to update the relative belief are given by $\boldsymbol{\theta} = (\theta_\Omega, \theta_\delta)^\top$, where the position of $\Omega_{\mathbf{y}}$ is encoded as an abscissa $\theta_\Omega \in [0.4, 26]$ m running along the structure axis. The MCMC analyses are carried out with batches of $N_{\text{obs}} = 8$ HF observations, all related to the same damage location θ_Ω and damage magnitude δ. However, each observation in a batch is obtained for random values of train velocity v and axle-carried mass m_A, which are presumed to be accurately measured by the train on-board system. Therefore, the relative posterior distribution is deterministically set to the measured values.

Tab. 7.3 summarizes the results of six MCMC simulations related to the sampling of $p(\boldsymbol{\theta}|\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}})$ for different operational and damage conditions. The damage location θ_Ω is consistently identified with high accuracy, except in Case 2. Nevertheless,

Case	Target($\theta_\Omega; \theta_\delta$)	Mean($\theta_\Omega; \theta_\delta$)	Mode($\theta_\Omega; \theta_\delta$)	StdDev($\theta_\Omega; \theta_\delta$)	L_{chain}
1	2.31 m; 73.42%	2.15 m; 70.96%	2.00 m; 75.00%	0.81 m; 7.03%	2650
2	3.96 m; 63.75%	2.38 m; 56.28%	2.30 m; 56.50%	0.38 m; 8.43%	2000
3	6.07 m; 47.53%	5.72 m; 50.91%	5.75 m; 41.5%	0.18 m; 8.37%	2000
4	9.44 m; 51.41%	9.19 m; 49.33%	9.15 m; 50.5%	0.72 m; 5.35%	2000
5	13.37 m; 41.25%	13.06 m; 43.97%	13.40 m; 41.50%	0.84 m; 3.95%	2000
6	17.13 m; 52.07%	16.27 m; 48.02%	16.20 m; 41.50%	1.45 m; 8.31%	2150

Table 7.3: Railway bridge - Damage localization and quantification outcomes for enhanced MCMC simulations under varying operational and damage conditions.

the relative discrepancy between the target and the posterior mean values is only 1.58 m over an admissible support of 25.6 m. Similarly, for the damage level θ_δ, the most significant deviation occurs in Case 2, with a discrepancy between the target and the posterior mean values of about 7.5%. An exemplary MCMC outcome is reported in Fig. 7.11 for Case 4.

7.3 CONCLUSIONS

In this Chapter, we have proposed a hybrid data/model approach to enhance the MCMC sampling of SHM parameters through a feature extractor and a feature-oriented SM. The feature extractor automates the extraction of informative features from raw sensor recordings, following the deep metric learning framework proposed in Chapter 5. The feature-oriented surrogate approximates the functional relationship between the parametric input space, for which we seek to update the relative belief, and the low-dimensional feature space. During the online monitoring phase, the two models are employed to evaluate the likelihood function underlying an MCMC sampler. The procedure leverages a preliminary offline phase in which: (i) we employ physics-based models and reduced-order modeling to overcome the lack of experimental data; (ii) we exploit the MF-DNN-SM proposed in Chapter 6 to efficiently generate a large labeled dataset; and (iii) we pre-process the sensor recordings via MTF encoding to transform them into images.

In comparison to the SHM setup presented in the previous Chapters, the proposed strategy offers several improvements. While the data-driven approach discussed in Chapter 5 was unable to account for relevant uncertainty sources, and the model-based approach detailed in Chapter 6 dealt with high-dimensional data hampering the inference of low-sensitivity damage parameters, the hybrid approach presented here incorporates both data-driven and model-based components to enable accurate real-time damage identification with quantified uncertainty. The data-driven aspect arises from the use of learnable features in place of raw data, allowing for damage identification accuracy comparable to that achieved in Chapter 5. Simultaneously, the model-based approach underlying the prob-

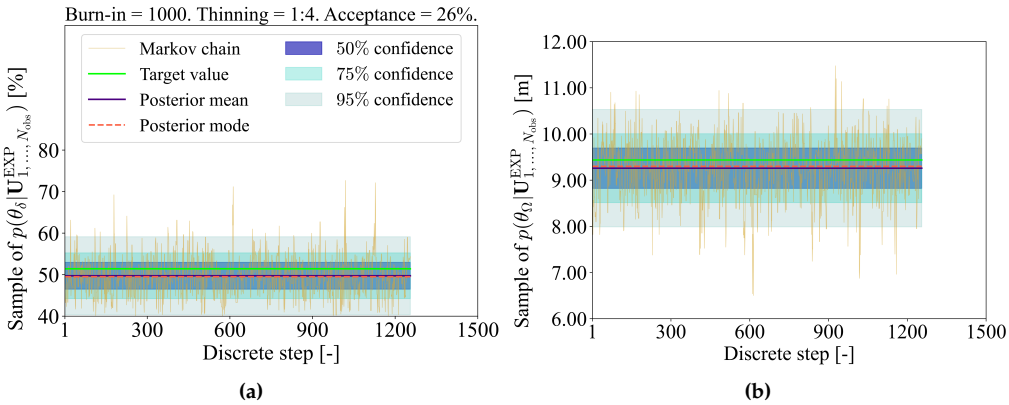


Figure 7.11: Railway bridge - Exemplary enhanced MCMC outcome (Case 4): Markov chain, target value, posterior mean, posterior mode, and credibility intervals relevant to (a) damage level θ_δ; (b) damage position θ_Ω.

abilistic MCMC framework maintains a measure of the reliability of the estimates provided, and it greatly benefits from DL-based feature extraction in terms of computational efficiency.

The proposed strategy has been assessed through the simulated monitoring of an L-shaped cantilever beam, a portal frame, and a railway bridge. A comparison with the MCMC outcomes presented in Sec. 6.3.3 has demonstrated several advantages of replacing $\mathrm{NN}_{\mathrm{MF}}$ with $\mathrm{NN}_{\mathrm{ENC}}$ and $\mathrm{NN}_{\mathrm{SUR}}$: (i) competitive computational costs due to the low dimensionality of the involved features and the efficiency of the DL models; (ii) improved convergence rates thanks to the geometrical structure characterizing the feature space; (iii) accurate estimates thanks to the informativeness of the extracted features.

The methodology can be easily adapted to solve inverse problems in application domains beyond SHM, such as material characterization, medical imaging, and seismic imaging. Although we have shown how to extract informative features in a low-dimensional metric space in the case of image data, data of diverse natures can be similarly addressed by selecting appropriate DL architectures.

Regarding the limitations, these directly stem from using the deep metric learning framework and the MF-DNN-SM. On one hand, there are concerns due to the uncertainties linked to the modeling assumptions, which pose a risk to the generalization capabilities of the strategy. On the other hand, there is a necessity to use potentially time-demanding HF numerical models to populate informative enough training datasets. Nevertheless, the obtained results demonstrate that a larger discrepancy between the target and the posterior mean values corresponds to an increase in the size of the confidence intervals, showcasing the reliability of the estimates provided. Moreover, the MF-DNN-SM partially mitigates the computational cost of supervised training.

8 DIGITAL TWINS FOR OPTIMAL MANAGEMENT AND MAINTENANCE OF STRUCTURES VIA PROBABILISTIC GRAPHICAL MODELS

Part of the research presented in this Chapter has been published as

[239] *Torzoni et al., A digital twin framework for civil engineering structures,* Computer Methods in Applied Mechanics and Engineering, *Vol. 418, 2024, p. 116584.*

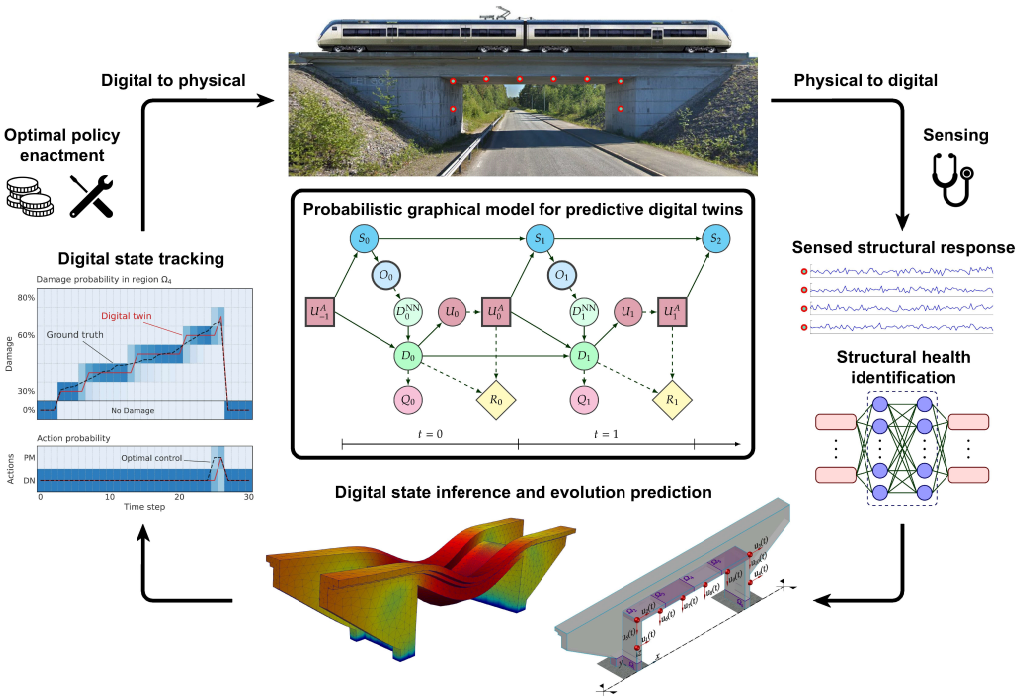
[347] *Torzoni et al., Digital-twin-SHM,* Github, *2024.*

19. Kapteyn *et al., A probabilistic graphical model foundation for enabling predictive digital twins at scale.* 2021

This Chapter is devoted to the development of a *DT framework* for health monitoring, maintenance, and management planning of engineering structures. The strategy is based on the foundational PGM proposed in [19], which provides a general framework for *data assimilation, state estimation, prediction, planning, and learning*. This PGM relies on a DDN, such as those introduced in Sec. 4.3.5, that encodes the end-to-end information flow, from physical to digital through *assimilation and inference*, and back to the asset in the form of *informed control actions*, all with *quantified uncertainty*. Although the capabilities of health-aware DTs are showcased in the specific context of monitoring the structural integrity of civil structures, the applicability of the presented framework is general and can be adapted for various types of engineering systems.

A graphical abstract of the methodology is depicted in Fig. 8.1, showing a *physical-to-digital* information flow and a *digital-to-physical* information flow. This *observations-to-decisions interaction* repeats indefinitely over time and involves:

- *From physical to digital.* Vibration recordings are gathered from the physical system and assimilated with DL models to estimate the structural health in terms of *presence, location, and severity of damage*. This involves jointly exploiting the regression and classification perspectives introduced in Sec. 4.1. Observational data are first processed with a *DL classifier* to address damage detection and localization. Whenever damage is detected and localized, the observations are further processed with a *DL regressor* to estimate the magnitude of damage. Within a *sequential Bayesian inference* formulation, this first estimate is then employed together with a *transition model* describing the expected evolution of the system state to update a *digital state* encoding the health state of the structure.



- *From digital to physical.* The updated digital state is employed to forecast the *future evolution* of the physical system and the associated uncertainty, leveraging the predictive capabilities of DBNs. The future digital state trajectories then guide *predictive decision-making* about maintenance and management actions within an MDP setting.

Figure 8.1: Predictive digital twin framework for civil engineering structures: graphical abstraction of the end-to-end information flow enabled by the probabilistic graphical model.

Moreover, we enhance the computational efficiency of the DT through a *preliminary offline learning phase*. This phase involves *training the DL models* underlying the structural health identification, and learning the *health-dependent control policy* to map the belief over the digital state onto actions that affect the asset. The DL models are trained in a supervised fashion, using data related to specific damage conditions generated with physics-based numerical models. Similarly to Chapter 5, we employ *reduced-order modeling* to efficiently assemble a sufficiently representative training dataset. Additionally, the control policy is computed by maximizing the expected future rewards for the planning problem induced by the PGM, employing the *value iteration algorithm* presented in Algorithm 4.1.

The novelty of this methodology lies in the tailored adaptation of the *PGM digital twinning framework* for the health monitoring, maintenance, and management planning of engineering structures. Moreover, the *assimilation of observational data* leverages the capabilities of CNNs to automatically extract

damage-sensitive features from raw vibration recordings and assess damage in real time. This data-driven capability enables the integration of high-dimensional sensor measurements into the DT framework while tracking the associated uncertainties.

The strategy is assessed against the same cantilever beam and railway bridge considered in the previous Chapters. In these case studies, we accurately track the evolution of structural health parameters and promptly suggest the most appropriate control input with relatively low uncertainty, demonstrating the *dynamic decision-making capabilities* of *health-aware DTs*.

The Chapter is organized as follows. Sec. 8.1 describes the PGM encoding the *asset-twin coupled dynamical system* and its use for structural health tracking, prediction, and planning. Sec. 8.2 explains the DL-based data assimilation for *digital state identification*. Two demonstrations of *dynamic data-driven health monitoring and asset management* related to a cantilever beam and a railway bridge are presented in Sec. 8.3 and Sec. 8.4, respectively. Conclusions are finally drawn in Sec. 8.5.

8.1 DIGITAL TWIN CONCEPT FOR ENGINEERING STRUCTURES

In this Section, we describe the methodology characterizing our DT framework in terms of the PGM encoding the asset-twin interaction together with the related sequential Bayesian inference formulation in Sec. 8.1.1, and of its extension to future prediction and optimal planning in Sec. 8.1.2.

8.1.1 MATHEMATICAL ENCODING OF ASSET-TWIN SYSTEMS

Similar to the SHM setup considered in Chapter 7, our DT assimilates vibration recordings shaped as multivariate time series $\mathbf{U}^{\text{EXP}} \in \mathbb{R}^{L \times N_u}$, consisting of N_u time series made of L sensor measurements equally spaced in time, under assumed steady operational and damage conditions.

The PGM defining the elements comprising the *asset-twin coupled dynamical system*, and mathematically describing the relevant *interactions through observed data and control inputs*, is the DDN sketched in Fig. 8.2. We adopt a *time discretization*, and denote non-dimensional time steps by t. The duration between consecutive time steps may vary depending on the application and is governed by the update frequency of the DT via data assimilation, ensuring the *DT is updated once per time step*.

In the graph, *circle nodes* denote random variables at discrete times, *square nodes* denote actions, and *diamond nodes* denote the

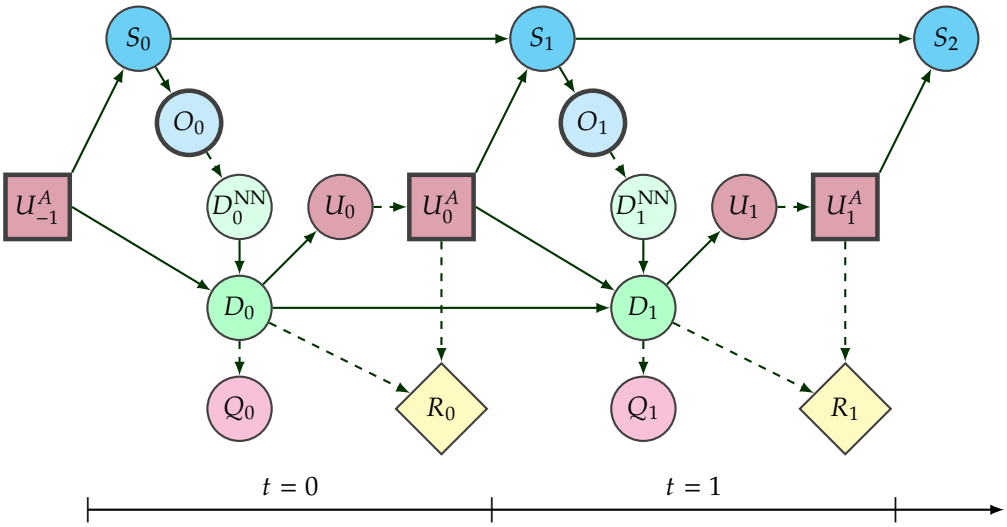


Figure 8.2: Dynamic decision network encoding the asset-twin coupled dynamical system. Circle nodes denote random variables, square nodes denote actions, and diamond nodes denote the objective function. Nodes with bold outlines denote observed quantities, while nodes with thin outlines denote unobserved quantities to be estimated. Directed solid edges represent the variables' dependencies encoded via conditional probability distributions, while directed dashed edges represent the variables' dependencies encoded via deterministic functions.

objective function. Nodes with *bold outlines* denote observed quantities, while nodes with *thin outlines* denote unobserved quantities to be estimated. The PGM is *sparsely connected* with edges encoding known or assumed *conditional dependencies*. *Solid edges* represent the variables' dependencies encoded via conditional probability distributions, modeled by CPTs as the spaces of unobserved variables are discrete. *Dashed edges* represent the variables' dependencies encoded via deterministic functions. Thanks to the modeled conditional dependencies, the *graph topology* is specified from the first two time slices and can be unrolled for every time step by repeating its core structure.

The unknown *physical state*, encapsulating the variability in the health state of the asset, is denoted as $S_t \sim p(s_t)$, where s_t represents the realization of the random variable S_t at time t, and $p(s_t)$ represents the probability that $S_t = s_t$, for any possible state s_t that the physical asset can assume.

The *digital state* $D_t \sim p(d_t)$ is characterized by a set of parameters that capture the variability of the physical asset by means of the computational model in Problem 3.1. This digital state may encompass various information, such as initial conditions, boundary conditions, material properties, and other relevant quantities for describing the asset. However, to yield maximal value, a DT does not necessarily need to be an exact virtual replica; instead, it should be designed to capture the *essential features and variations in the asset* that are pertinent for diagnosis, prediction, and decision-making in the application of interest [348]. In our case, we define an asset-specific digital state by selecting a specific subset of $\boldsymbol{\mu}$, which involves the parameters $\{\ell, \delta\}$ describing the *presence, location and magnitude of damage*

348. Ferrari *et al.*, *Digital twins in mechanical and aerospace engineering*. 2024

that can affect the asset among $N_{\mathcal{D}} \in \mathbb{N}$ possible configurations in the digital state space. In particular, we model damage as a *local reduction in material stiffness* and address the problem of *damage detection and localization from a classification perspective*, as discussed in Sec. 4.1.2. The parameter $\ell \in \{0, \dots, N_\ell\}$ represents the specific *damage region*, among a set of $N_\ell \in \mathbb{N}$ *predefined damage locations*, where $\ell = 0$ denotes the damage-free baseline. The parameter $\delta \in \mathbb{R}$ instead represents the *magnitude of the stiffness reduction* occurring within the region associated with ℓ and is addressed from a *regression perspective*.

The *physical-to-digital* information flow from S_t to D_t is governed by the *observed data* $O_t = o_t$. They may encompass sensor data, inspection data, or information from diagnostic systems. In our case, these data are the vibration recordings $\mathbf{U}^{\text{EXP}}$. Trough the edge $S_t \to O_t$, the physical state S_t is only *partially and indirectly observable*, meaning that the digital state D_t can not be uniquely determined. Accordingly, it only provides evidence for different digital state configurations that can be consistent with a given observation.

The observations are assimilated, as described in Sec. 8.2, to provide an *assimilated digital state* $D_t^{\text{NN}} \sim p(d_t^{\text{NN}})$ by solving the *SHM inverse problem*. The edge $O_t \to D_t^{\text{NN}}$ is dashed to denote the use of DL models in solving the SHM inverse problem. This initial estimation is then incorporated into a *sequential Bayesian inference formulation*, along with the *prior belief* D_{t-1} from the previous time step, to estimate an *updated digital state* D_t according to a *control-dependent transition model* describing the expected evolution of the digital state. This process is encoded by the conditional dependencies associated with $(D_{t-1}, U_{t-1}^A) \to D_t \leftarrow D_t^{\text{NN}}$.

The updated digital state can be exploited to compute *quantities of interest* $Q_t \sim p(q_t)$, following the edge $D_t \to Q_t$. These quantities may include outputs of the computational models comprising the DT, such as modal quantities or other response features. This is typically a *many-query task*, suitable to be addressed with reduced-order or surrogate modeling techniques, as discussed in Sec. 1.3.1. Quantities of interest can be useful to perform *posterior predictive checks* on the tracking capabilities of the DT by comparing sensor measurements with their posterior estimates. However, we stress that we do not exploit this capability here, and the Q_t node is retained in the graph only for consistency with the foundational model proposed in [19].

19. Kapteyn *et al.*, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*. 2021

The updated digital state D_t is exploited to inform the *digital-to-physical* information flow, i.e., the *control inputs*. In Fig. 8.2, both U_t and U_t^A represent *decision variables*. These actions

could involve interventions that affect the physical state (such as deciding to perform a maintenance work), the operational conditions of the asset (such as deciding to modify the loading configuration), or the observational data generated by the asset (such as deciding to perform an inspection or modifying the deployed sensing system). Following the setup described in Sec. 4.3.5, $U_t \sim p(u_t)$ is depicted with a circle node, representing the *predicted* belief about the action to take from a set of $N_{\mathcal{U}} \in \mathbb{N}$ possible actions, while $U_t^A = u_t^A$ is depicted with a square node, representing the *actual* control effectively taken. At each time step, U_t is estimated according to a *health-dependent control policy* that maps the belief over the digital state to the space of actions, following the edge $D_t \rightarrow U_t$. The dashed connection $U_t \rightarrow U_t^A$ instead denotes a *conversion from probabilistic to deterministic control* by selecting the best point estimate of U_t.

Finally, the *reward* $R_t \sim p(r_t)$ quantifies the performance of the asset for the time step and can be equivalently perceived as a negative cost. In particular, R_t measures the *expected utility associated with different trajectories of the asset-twin system*. The two dashed connections $D_t \rightarrow R_t \leftarrow U_t^A$ denote a *deterministic reward function* quantifying costs relative to health state and control inputs, respectively. Consistently with the setup described in Sec. 4.3.5, this reward serve as the quantity we seek to optimize when solving the *planning problem* for the control policy.

The PGM encodes two key assumptions: (i) the physical state is only observable indirectly via observational data; (ii) the physical and digital states evolve according to a Markov process. The resulting *conditional independence structure* allows us to frame the asset tracking within a sequential Bayesian inference framework. Indeed, by exploiting conditional independence and Bayes rule, the joint distribution over variables can be factorized from the initial time step, $t = 0$, up to the *current time*, $t = t_c$, as:

$$p(D_{0:t_c}^{\mathrm{NN}}, D_{0:t_c}, Q_{0:t_c}, R_{0:t_c}, U_{0:t_c} | O_{0:t_c} = o_{0:t_c}, U_{-1:t_c}^A = u_{-1:t_c}^A) \propto \prod_{t=0}^{t_c} \left[\phi_t^{\mathrm{data}} \phi_t^{\mathrm{update}} \phi_t^{\mathrm{QoI}} \phi_t^{\mathrm{control}} \phi_t^{\mathrm{reward}} \right], \tag{8.1}$$

with factors:

$$\phi_t^{\mathrm{data}} = p(O_t = o_t | D_t^{\mathrm{NN}}) p(D_t^{\mathrm{NN}}), \tag{8.2}$$

$$\phi_t^{\mathrm{update}} = p(D_t | D_{t-1}, U_{t-1}^A = u_{t-1}^A, D_t^{\mathrm{NN}}), \tag{8.3}$$

$$\phi_t^{\mathrm{QoI}} = p(Q_t | D_t), \tag{8.4}$$

$$\phi_t^{\mathrm{control}} = p(U_t | D_t), \tag{8.5}$$

$$\phi_t^{\mathrm{reward}} = p(R_t | D_t, U_t^A = u_t^A). \tag{8.6}$$

Here, ϕ_t^{data} encodes the assimilation of observed data through the DL models underlying the structural health identification. We assume $p(D_t^{\text{NN}})$ to be uniform, simplifying it as $\phi_t^{\text{data}} \propto p(O_t = o_t | D_t^{\text{NN}})$*. $\phi_t^{\text{update}} \in \mathbb{R}^{N_{\mathcal{D}} \times (N_{\mathcal{D}} N_{\mathcal{U}} N_{\mathcal{D}})}$ is a CPT that updates the digital state belief D_t by integrating information from multiple sources. It accounts for a *control-dependent predictor forward in time*, describing the expected evolution of the digital state through the conditioning on D_{t-1} and $U_{t-1}^A = u_{t-1}^A$, and integrates this prior with the *data assimilation outcome* through the conditioning on D_t^{NN}. The contribution encoding the *control-dependent transition dynamics* can be visualized as a CPT $\phi_t^{\text{history}} \in \mathbb{R}^{N_{\mathcal{D}} \times (N_{\mathcal{D}} N_{\mathcal{U}})}$, similar to that in Eq. (4.54). This CPT should embody any a-priori knowledge that the DT designer has regarding the asset and the relevant operational conditions. It can be estimated offline from historical data, see, e.g., [349,350,351], derived from physics-based or empirical damage progression models, see, e.g., [26,352], or learned online from experience. On the other hand, the *data assimilation* contribution takes the form of a CPT $\phi_t^{\text{NN}} \in \mathbb{R}^{N_{\mathcal{D}} \times N_{\mathcal{D}}}$, mapping the estimate D_t^{NN} provided by the DL models onto a belief over the digital state space. This CPT is a *confusion matrix*† that measures the offline (expected) performance of the DL models in correctly identifying the digital state among the $N_{\mathcal{D}}$ possible outcomes of D_t. For $t = 0$, ϕ_t^{update} simplifies to $\phi_{t=0}^{\text{update}} = p(D_{t=0} | U_{t=-1}^A = u_{t=-1}^A, D_{t=0}^{\text{NN}})$ to accommodate the initial distribution over random variables. ϕ_t^{QoI} encodes the computation of quantities of interest using the updated digital state, while ϕ_t^{reward} evaluates the reward function quantifying the performance of the asset. Finally, the control factor $\phi_t^{\text{control}} \in \mathbb{R}^{N_{\mathcal{U}} \times N_{\mathcal{D}}}$ is a CPT encoding the health-dependent control policy. This control policy is computed by optimizing R_t, as described in the next section. Since the spaces of the unobserved variables are discrete, we can propagate and update the relative belief exactly with a single pass of the *sum-product message-passing algorithm* [326]‡.

*It is worth noting that, in our case, the use of the Bayes rule for ϕ_t^{data} is concerned with the application of deep learning algorithms to solve the inverse problem associated with the identification of the structural health. This involves approximating the inverse mapping from measurements to sought parameters. Accordingly, we consider assuming a uniform prior for the assimilated digital state to be valid if the deep learning models are not biased towards predicting a certain digital state over others. The deep learning models are not biased if, as in our case, the training dataset is balanced among the possible digital state outcomes. Indeed, the training data are generated by taking the input parameters as uniformly distributed within their value ranges, allowing to tune the deep learning models on loss function contributions evenly distributed among the possible digital state outcomes.

349. Morcous, *Performance Prediction of Bridge Deck Systems Using Markov Chains*. 2006

350. Thompson *et al.*, *Markovian bridge deterioration: developing models from historical data*. 2005

351. Bianchi *et al.*, *Life-Cycle Assessment of Deteriorating RC Bridges Using Artificial Neural Networks*. 2022

26. Biondini *et al.*, *Life-Cycle Performance of Deteriorating Structural Systems under Uncertainty: Review*. 2016

352. Biondini *et al.*, *Time-variant redundancy and failure times of deteriorating concrete structures considering multiple limit states*. 2017

†A "confusion matrix" is a table to describe the performance of a classification model against labeled testing data. In our confusion matrix, rows represent the actual classes, while columns represent the predicted classes. Each cell in the matrix collects the percentage of instances that are classified correctly or incorrectly.

326. Pearl, *Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach*. 1982

‡Note that R_t is computed using continuous functions, taking inputs from the discrete sets of digital states and controls.

8.1.2 Extension to Prediction and Optimal Planning

Starting from the updated belief state at the current time step t_c, future predictions of D_t and U_t are achieved though the DBN in Fig. 8.3. This *prediction PGM* is obtained by unrolling the portion of PGM relative to D_t, Q_t, R_t, and U_t, until a *prediction time* t_p. All other nodes are removed from the prediction PGM, as neither data are assimilated nor actions are performed while forecasting the evolution of D_t and U_t. The factorization in

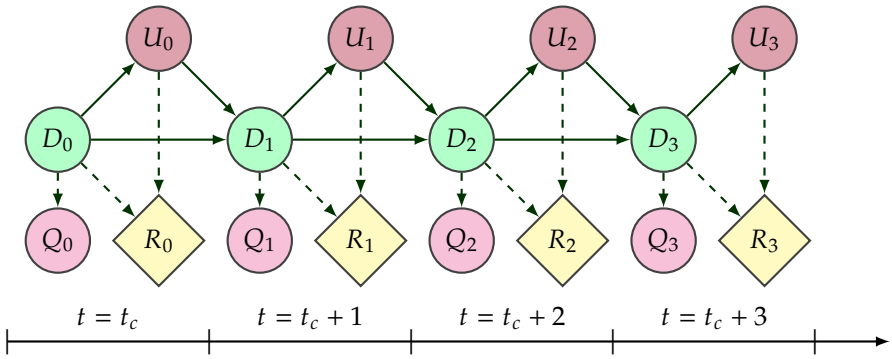


Figure 8.3: Dynamic Bayesian network employed to predict the future evolution of digital states and control inputs. Circle nodes denote random variables, and diamond nodes denote the objective function. Directed solid edges represent the variables' dependencies encoded via conditional probability distributions, while directed dashed edges represent the dependencies encoded via deterministic functions.

Eq. (8.1) can be extended over the *prediction horizon* associated with t_p by augmenting the belief state as:

$$p(D_{t_c+1:t_p}, Q_{t_c+1:t_p}, R_{t_c+1:t_p}, U_{t_c+1:t_p} | o_{0:t_c}, u^A_{-1:t_c}) \propto \prod_{t=t_c+1}^{t_p} \left[\phi_t^{\text{future-}D} \phi_t^{\text{QoI}} \phi_t^{\text{control}} \phi_t^{\text{future-}R} \right], \tag{8.7}$$

where, similarly to Eq. (4.60), $\phi_t^{\text{future-}D} = p(D_t | D_{t-1}, U_{t-1})$ and $\phi_t^{\text{future-}R} = p(R_t | D_t, U_t)$ are identical to ϕ_t^{history} and ϕ_t^{reward}, respectively, but they are conditioned on *predicted control inputs* that have not yet been selected and applied.

The algorithmic description of the online phase of the proposed DT framework is reported in Algorithm 8.1. The operations, encoded by the interaction factors in Eq. (8.2)-Eq. (8.6), repeat each time new observational data are acquired to estimate the unobserved variables. We first increment the current time step t_c and add a new slice of nodes to the *online phase PGM*. Observational data $O_{t_c} = o_{t_c}$ are assimilated with the DL models (ϕ_t^{data}). Subsequently, the assimilated digital state $D_{t_c}^{\text{NN}}$ is employed to update D_{t_c-1} to D_{t_c} and estimate the control inputs U_{t_c} (ϕ_t^{update}, ϕ_t^{control}). The updated digital state is then propagated using the *prediction PGM* to compute a probabilistic estimate of future digital states $D_{t_c+1:t_p}$ and controls $U_{t_c+1:t_p}$ ($\phi_t^{\text{future-}D}$, ϕ_t^{control}). Finally, the actual control input $U_{t_c}^A = u_{t_c}^A$ is selected as the best point estimate of U_{t_c}.

The ϕ_t^{reward} factor, or equivalently $\phi_t^{\text{future-}R}$, is implicitly embedded in the control policy $\pi(D_t)$ ruling ϕ_t^{control}. $\pi(D_t)$ is computed offline under the simplifying assumption of a *sufficient sensing capability to provide an accurate estimate of the structural health*. This assumption entails having a fully observable digital state, which decouples the sensing and control problems. This allows for planning an *optimal course of actions* $\pi^*(D_t)$ in response to the expected evolution of structural health by solving the MDP associated with the prediction PGM. Considering an

Algorithm 8.1 Digital twin framework of engineering structures. Online phase: algorithmic description

Input: topology and factors for the online phase PGM
topology and factors for the prediction PGM
trained DL models
observational data $O_{t_c} = o_{t_c}$
action enacted at previous time step $U^A_{t_c-1} = u^A_{t_c-1}$

1: assimilate o_{t_c} with DL models to provide $D^{\text{NN}}_{t_c}$
▹ $(O_{t_c}) \longrightarrow (D^{\text{NN}}_{t_c})$
2: infer D_{t_c}, U_{t_c}, given $D_{t_c-1}, u^A_{t_c-1}, D^{\text{NN}}_{t_c}, \phi^{\text{update}}_t, \phi^{\text{control}}_t$
▹ $(D_{t_c-1}, D^{\text{NN}}_{t_c}, U^A_{t_c-1},) \longrightarrow (D_{t_c}, U_{t_c})$
3: infer future evolution of D_t, U_t, given $D_{t_c}, \phi^{\text{future-}D}_t, \phi^{\text{control}}_t$
▹ $(D_{t_c}) \longrightarrow (D_{t_c+1:t_p}, U_{t_c+1:t_p})$
4: select $U^A_{t_c} = u^A_{t_c}$ as the best point estimate of U_{t_c}
▹ $(U_{t_c} \longrightarrow (U^A_{t_c})$
5: **return** control input to be enacted $U^A_{t_c} = u^A_{t_c}$,
6: expected evolution $p(D_{t_c+1:t_p})$,
7: expected evolution $p(U_{t_c+1:t_p})$

infinite planning horizon, this can be formulated as the following optimization problem:

$$\pi^*(D_t) = \arg\max_{\pi} \sum_{t=0}^{+\infty} \gamma^t \mathbb{E}[R_t], \tag{8.8}$$

which is solved through the *value iteration method* described in Algorithm 4.1 using a *discount factor* $\gamma \in (0, 1]$. Similar to Eq. (4.59), the reward function to be optimized is chosen as:

$$R_t(U_t, D_t) = R^{\text{control}}_t(U_t) + \chi_R R^{\text{health}}_t(D_t), \tag{8.9}$$

where $R^{\text{control}}_t(U_t)$ and $R^{\text{health}}_t(D_t)$ quantify the rewards relative to control inputs and health state, respectively, and $\chi_R \in \mathbb{R}$ is a weighting factor to tune the trade-off between *risk-averse* and *risk-seeking* behavior. In general, the reward values might represent real-world quantities such as costs, or they could be more abstract quantities tuned to achieve the desired system performance. For instance, R^{health}_t can resemble a variety of aspects, like reduction in the level of service due to deterioration, workplace accidents, structural reliability, and structural failure probability [319].

Both the online phase PGM and the prediction PGM have been implemented in the `Python` environment, using the `pgmpy` library [353]. The DT framework is general, and can easily be adapted to deal with various applications by reorganizing the topology of the graph, if needed.

319. Papakonstantinou *et al.*, *Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation.* 2014

353. Ankan *et al.*, *pgmpy: Probabilistic graphical models using Python.* 2015

8.2 DATA ASSIMILATION VIA ARTIFICIAL NEURAL NETWORKS

The ϕ_t^{data} factor encodes the assimilation of observed data through the DL models underlying the structural health identification. The DL models are trained exploiting a *simulation-based strategy*. By following the procedure detailed in Sec. 3.2, the training dataset is populated by taking $\boldsymbol{\mu}$ as uniformly distributed and sampling it via Latin hypercube rule, to provide:

$$\mathcal{D}_{\text{DT}} = \{(\mathbf{U}_i, \mathscr{E}_i, \delta_i)\}_{i=1}^{N_{\text{train}}}, \tag{8.10}$$

collecting N_{train} vibration recordings generated through a reduced basis ROM. Sensor recordings $\mathbf{U}_i$ are extracted using Eq. (3.23) and labeled with the corresponding values of $\mathscr{E}_i$ and δ_i. In order to mimic measurement noise, each vibration recording in $\mathcal{D}_{\text{DT}}$ is corrupted by adding an independent, identically distributed Gaussian noise. In the following, the index i will be dropped for ease of notation, unless necessary. As for the previous Chapters, the FOMs and ROMs underlying Problem (3.1) and Problem (3.18) have been implemented in the `Matlab` environment, using the `redbKIT` library [330]. All computations have been carried out on the same personal computer featuring an `AMD Ryzen™ 9 5950X` CPU @ 3.4 GHz and 128 GB RAM.

330. Negri, *redbKIT, version 2.2.* 2016

In the remainder of this Section, we describe the adopted DL models in Sec. 8.2.1, and how they are used within the DT for detecting, locating, and quantifying the presence of structural damage in Sec. 8.2.2.

8.2.1 DAMAGE IDENTIFICATION

Every time new observational data are acquired, they are first processed with a *classification model* NN_{CL} to address damage detection and localization. Here, the classes are those described through the $\mathscr{E}$ parameter. Whenever damage is detected and localized in the m-th region, with $m = 1, \dots, N_{\mathscr{E}}$, the observational data are further processed with a *regression model* NN_{RG}^m — one for each damageable region — to quantify the associated amount of damage δ.

The model NN_{CL} addresses the multi-class classification task underlying the damage detection/localization problem, namely $\text{NN}_{\text{CL}} : \mathbb{R}^{L \times N_u} \mapsto \mathbb{B}^{N_{\mathscr{E}}+1}$, $\mathbf{U} \mapsto \mathbf{b}$. Following the setup described in Sec. 4.1.2, $\mathbf{b} = [b^0, \dots, b^{N_{\mathscr{E}}}]^\top$ is a *one-hot encoding*, with entries b^n equal to 1 if the target class $\mathscr{E}$ is n and 0 otherwise, with $n = 0, \dots, N_{\mathscr{E}}$, categorizing one of the $N_{\mathscr{E}} + 1$ predefined damage scenarios described through $\mathscr{E}$. By employing a *Softmax*

activation function for the output layer of $\mathrm{NN}_{\mathrm{CL}}$, see Eq. (4.9), the entries of the $\mathrm{NN}_{\mathrm{CL}}$ output $\widehat{\mathbf{b}} = (\widehat{b}^0, \ldots, \widehat{b}^{N_\mathscr{b}})^\top \in \mathbb{R}^{N_\mathscr{b}+1}$ are interpreted as the confidence levels by which $\mathbf{U}$ is assigned to each damage class.

The training of $\mathrm{NN}_{\mathrm{CL}}$ is carried out using a dedicated dataset, derived from $\mathcal{D}_{\mathrm{DT}}$ in a similar manner to Eq. (4.4), as follows:

$$\mathcal{D}_{\mathrm{CL}} = \{(\mathbf{U}_i, \mathbf{b}_i)\}_{i=1}^{N_{\mathrm{train}}}, \tag{8.11}$$

which serves to optimize the set of weights $\mathbf{\Omega}_{\mathrm{CL}}$ parametrizing $\mathrm{NN}_{\mathrm{CL}}$. The optimization is carried out by minimizing the probabilistic *categorical cross-entropy* loss function between the predicted and target class labels over $\mathcal{D}_{\mathrm{CL}}$, similar to Eq. (4.8):

$$\mathcal{L}_{\mathrm{CL}}(\mathbf{\Omega}_{\mathrm{CL}}, \mathcal{D}_{\mathrm{CL}}) = -\frac{1}{N_{\mathrm{train}}} \sum_{i=1}^{N_{\mathrm{train}}} \sum_{n=0}^{N_\mathscr{b}} b_i^n \log(\widehat{b}_i^n), \tag{8.12}$$

providing a measure of the dissimilarity between the one-hot encoding $\mathbf{b}$ and its estimated counterpart $\widehat{\mathbf{b}}$.

For any given set of input vibration recordings $\mathbf{U}^{\mathrm{EXP}}$, the trained $\mathrm{NN}_{\mathrm{CL}}$ provides the estimated counterpart of $\mathbf{b}$ as:

$$\widehat{\mathbf{b}} = \mathrm{NN}_{\mathrm{CL}}(\mathbf{U}^{\mathrm{EXP}}), \tag{8.13}$$

and the most likely class is selected as the best point estimate categorizing the processed measurements $\mathbf{U}^{\mathrm{EXP}}$.

The model $\mathrm{NN}_{\mathrm{RG}}^m$ addresses the regression task underlying the damage quantification problem, namely $\mathrm{NN}_{\mathrm{RG}}^m : \mathbb{R}^{L \times N_u} \mapsto \mathbb{R}$, $\mathbf{U} \mapsto \delta$, with $m = 1, \ldots, N_\mathscr{b}$. Each regression model maps the vibration recordings $\mathbf{U}$ associated with the m-th damage region onto the estimated magnitude of the stiffness reduction occurring within that damage region. Since all these regression models are learned following the same procedure, the index m will be dropped in the following for ease of notation.

The dataset dedicated to train $\mathrm{NN}_{\mathrm{RG}}$ is derived from $\mathcal{D}_{\mathrm{DT}}$, as follows:

$$\mathcal{D}_{\mathrm{RG}} = \{(\mathbf{U}_{i_{\mathrm{RG}}}, \delta_{i_{\mathrm{RG}}})\}_{i_{\mathrm{RG}}=1}^{N_{\mathrm{train}}^{\mathrm{RG}}}, \tag{8.14}$$

where $N_{\mathrm{train}}^{\mathrm{RG}} \in \mathbb{N}$ is the number of training instances in $\mathcal{D}_{\mathrm{RG}}$, all characterized by a structural damage within the same region.

The set of weights $\mathbf{\Omega}_{\mathrm{RG}}$ parametrizing $\mathrm{NN}_{\mathrm{RG}}$ is optimized by minimizing the following MSE loss function:

$$\mathcal{L}_{\mathrm{RG}}(\mathbf{\Omega}_{\mathrm{RG}}, \mathcal{D}_{\mathrm{RG}}) = \frac{1}{N_{\mathrm{train}}^{\mathrm{RG}}} \sum_{i_{\mathrm{RG}}=1}^{N_{\mathrm{train}}^{\mathrm{RG}}} (\delta_{i_{\mathrm{RG}}} - \widehat{\delta}_{i_{\mathrm{RG}}})^2, \tag{8.15}$$

measuring the distance between the target magnitude of the stiffness reduction δ and its estimated counterpart $\widehat{\delta}$.

Algorithm 8.2 Digital twin framework of engineering structures. Preliminary offline phase: algorithmic description

Input: parametrization of the physics-based models
POD tolerance ϵ_σ
topology of the online phase PGM
topology of the prediction PGM
training options for the classifier $\mathrm{NN}_{\mathrm{CL}}$
training options for the regressors $\mathrm{NN}_{\mathrm{RG}}$
discount factor γ
tolerance on the value error for any state $\epsilon_{\mathbb{V}}$

set up the numerical model of the structure
assemble the snapshot matrix $\mathbf{S}$ via FOM analyses
compute the POD basis matrix via SVD of $\mathbf{S}$
use the ROM to populate the training datasets $\mathcal{D}_{\mathrm{CL}}$ and $\mathcal{D}_{\mathrm{RG}}$
train $\mathrm{NN}_{\mathrm{CL}}$ on $\mathcal{D}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ on $\mathcal{D}_{\mathrm{RG}}$
test $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ against noisy FOM data
compute the CPT encoding ϕ_t^{NN}
set the CPT encoding $\phi_t^{\mathrm{history}}$
solve the MDP induced by prediction PGM (value iteration)
set the reward function encoding ϕ_t^{reward}
return trained classifier $\mathrm{NN}_{\mathrm{CL}}$,
trained regressors $\mathrm{NN}_{\mathrm{RG}}$,
CPT encoding ϕ_t^{NN},
CPT encoding $\phi_t^{\mathrm{history}}$,
control policy $\pi(D_t)$ encoding $\phi_t^{\mathrm{control}}$,
reward function encoding ϕ_t^{reward}

Given a set of vibration recordings $\mathbf{U}^{\mathrm{EXP}}$ for which damage has been detected and localized by means of $\mathrm{NN}_{\mathrm{CL}}$, the trained $\mathrm{NN}_{\mathrm{RG}}$ estimates δ for the identified damage region, as follows:

$$\widehat{\delta} = \mathrm{NN}_{\mathrm{RG}}(\mathbf{U}^{\mathrm{EXP}}). \tag{8.16}$$

The DL architectures of $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ have been implemented through the `Tensorflow`-based `Keras` API [250]. The relevant implementation details are reported in Appendix E.

250. Chollet *et al.*, *Keras*. 2015

8.2.2 DIGITAL STATE IDENTIFICATION

Since the space of digital states is discrete, the outcomes of $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ are accommodated within the PGM by discretizing the value range of δ into $N_\delta \in \mathbb{N}$ intervals. The number and width of the intervals are chosen arbitrarily, and there are no restrictions in this regard. The resulting $N_{\mathcal{D}} = 1 + N_\delta N_{\mathscr{E}}$ digital states are sorted first by damage location and then by damage level. The algorithmic description of the offline phase characterizing the proposed DT framework is reported in Algorithm 8.2.

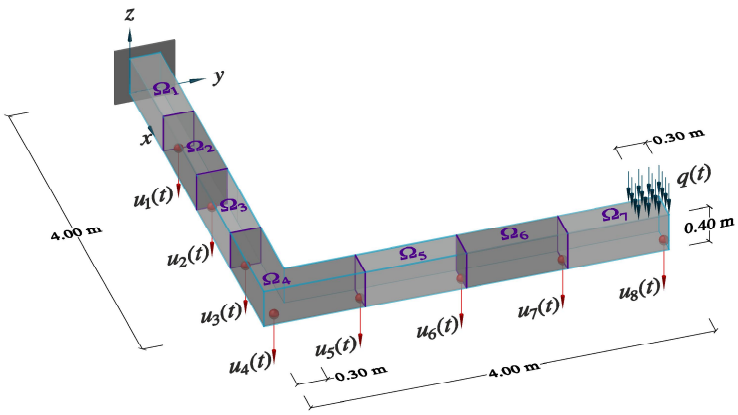


Figure 8.4: L-shaped cantilever beam: details of synthetic recordings related to displacements $u_1(t), \ldots, u_8(t)$, loading condition, and predefined damage regions $\Omega_1, \ldots, \Omega_7$.

8.3 DEMONSTRATION: L-SHAPED CANTILEVER BEAM

The first test case deals with the L-shaped beam depicted in Fig. 8.4. The SHM setup is similar to that previously considered in Sec. 5.3, Sec. 6.2.1, and Sec. 7.2.1.

Synthetic displacement time histories $\mathbf{U}$ over $(0, T = 1 \text{ s})$ are recorded by means of $N_u = 8$ DOFs with an acquisition frequency of $f_s = 200$ Hz, to mimic the monitoring system depicted in Fig. 8.4. The recordings are corrupted with an additive Gaussian noise to yield an SNR of 100.

The sinusoidal load is parametrized with $Q \in [40, 80]$ kPa and $f_q \in [10, 60]$ Hz. In addition to the damage-free condition, damage is simulated by considering $N_\mathscr{E} = 7$ possible damage classes. Each class refers to a reduction of the material stiffness within a subdomain Ω_m, with $m = 1, \ldots, N_\mathscr{E}$, as depicted in Fig. 8.4. The stiffness reduction can occur with a magnitude $\delta \in [30\%, 80\%]$, and it is held constant within $(0, T)$.

8.3.1 DATASET ASSEMBLY

The FOM features $N_{\text{FE}} = 4659$ DOFs, while the basis matrix $\mathbf{W}$ is derived from a snapshot matrix $\mathbf{S}$ assembled through $N_P = 400$ FOM evaluations, at varied values of $\boldsymbol{\mu} = (Q, f_q, \mathscr{E}, \delta)^\top$. Prescribing a tolerance $\epsilon_\sigma = 10^{-3}$ in Eq. (3.16), the ROM order is set to $N_{\text{RB}} = 56$. By taking $\boldsymbol{\mu}$ as uniformly distributed and sampling it via Latin hypercube rule, we use the ROM to populate dataset $\mathcal{D}_{\text{DT}}$ with $N_{\text{train}} = 10,000$ instances. This dataset is then employed to train NN_{CL} and NN_{RG}, as described in Sec. 8.2.1.

8.3.2 DIGITAL TWIN FRAMEWORK

The testing phase of NN_{CL} and NN_{RG} is carried out through noise-corrupted FOM solutions. In particular, the asset is monitored by processing batches of $N_{\text{obs}} = 10$ observations. Each

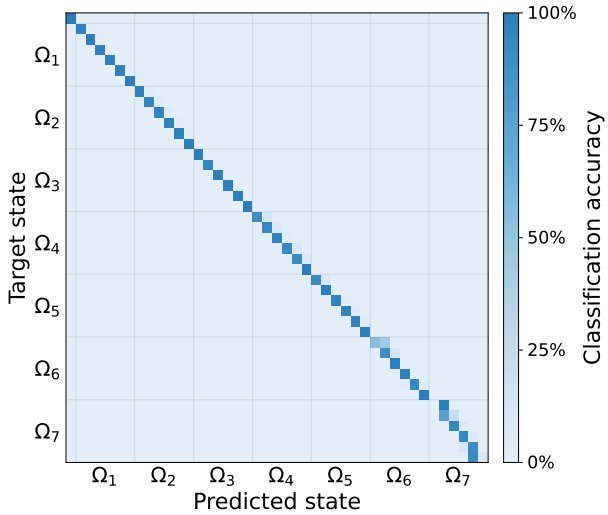


Figure 8.5: L-shaped cantilever beam - Confusion matrix measuring the offline performance of the DL models in correctly categorizing the digital state. Results are reported in terms of classification accuracy, measuring how observational data are classified with respect to the ground truth digital state. Digital states are ordered first by damage location and then by damage level.

observation in the batch is relative to the same $\mathscr{E}$ and δ, while Q and f_q are randomly sampled from their value ranges. Each time the DT assimilates a batch of noisy observations $\mathbf{U}^{\text{EXP}}_{1,\dots,N_{\text{obs}}}$, it dynamically estimates the variation in the structural health parameters underlying the digital state.

In order to accommodate the outcome of NN_{CL} and NN_{RG} into the PGM and to derive the ϕ_t^{NN} factor, the range in which δ can take values is discretized into $N_\delta = 6$ intervals: $\{[30\%, 35\%], [35\%, 45\%], [45\%, 55\%], [55\%, 65\%], [65\%, 75\%], [75\%, 80\%]\}$. The resulting $N_{\mathcal{D}} = 43$ possible digital states are then sorted first by damage location and then by damage level.

The confusion matrix reported in Fig. 8.5 encapsulates the ϕ_t^{NN} factor. This matrix quantifies the offline performance of NN_{CL} and NN_{RG} in assimilating noisy FOM data to classify the digital state. This assessment is carried out using $N_{\text{test}} = 4000$ testing instances, resulting in 500 data points for each damage class. We first discretize the value range of δ into equally spaced 1000 query points. Then, by looping over the query δ values for each damage class, we scan the N_{test} instances and retrieve the N_{obs} closest ones in terms of δ to categorize the digital state. We note that discretizing the value range of δ does not impact the instances associated with $\mathscr{E} = 0$ in terms of damage. However, it enables picking structural responses associated with varying operational conditions, as for the other cases relative to $\mathscr{E} \neq 0$. Regarding the performance, the ground truth digital state is detected with a classification accuracy of 93.61%. Most of the misclassifications stem from damage scenarios involving Ω_6 or Ω_7. This outcome aligns with what observed in Sec. 5.3, Sec. 6.2.1, and Sec. 7.2.1, with measurements close to

the clamped side only marginally affected by the presence of damage close to the free end.

For the present case, we consider four possible control inputs, each provided with a CPT modeling the *transition probability* $p(D_t|D_{t-1}, U^A_{t-1} = u^A_{t-1})$, and collectively encoding the ϕ_t^{history} factor. Note that these internal models do not reflect the ground truth evolution of structural health, which remains unknown to the DT. The considered control inputs are as follows:

I *Do nothing* (DN) action. No maintenance action is performed, and the physical state evolves following a stochastic deterioration process.

II *Perfect maintenance* (PM) action. A maintenance action is performed, restoring the asset from its current condition to the damage-free state. This can be traced back to replacing excessively compromised structural elements.

III *Minor imperfect maintenance* (MI) action. A maintenance action is performed, potentially restoring the asset from its current condition to a healthier damage state. This can be traced back to, e.g., patching and sealing cracked surfaces, rectifying and replacing expansion joints, or tightening/replacing loose/missing knot bolts for steel members.

IV *Major imperfect maintenance* (MA) action. A maintenance action is performed, with a higher probability of restoring the asset to a healthier damage state compared to the MI action. This can be traced back to, e.g., repairing heavily damaged slabs, piers, and steel members, and retrofitting compromised structural elements.

8.3.3 Results: two available actions

We first demonstrate the DT capabilities to assimilate observational data and track the structural health evolution by limiting the available actions to DN and PM. The (unknown) ground truth evolution of structural health varies depending on the most recent control input. In the absence of maintenance actions, this evolution is governed by a prescribed (simulated) stochastic *deterioration process* that monotonically deteriorates the structural health. This degradation process features a probability of damage inception ($\ell \neq 0$) equal to 0.5. Damage may develop in any of the N_ℓ predefined regions with $\delta = 30\%$, and then propagate with δ increments sampled from a Gaussian PDF centered at 1.5% with a standard deviation equal to 1% (negative increments are rounded to zero). The prescribed trajectory of

the structural health parameters is arbitrarily chosen to fully display the capabilities of the DT. Nevertheless, the DT would be equally capable of tracking the structural health evolution considering either more or less aggressive degradation processes, as demonstrated in the next section. The effect of a PM action is instead simulated by restoring the physical state to the undamaged configuration. At each time step, new observational data are simulated according to the (unknown) ground truth structural health and the most recently enacted control input. The DT assimilates these data, estimates the digital state, and suggests the next control input.

The state transition model encoding ϕ_t^{history} collects transition probabilities $p(D_t|D_{t-1}, U_{t-1}^A = u_{t-1}^A)$, conditioned on the most recently issued control input. For each action $U_{t-1}^A = u_{t-1}^A$, the diagonal entries of the associated CPT represent the probability of remaining in the same state. The lower-left and upper-right triangles are instead associated with the probabilities of the system deteriorating and improving its condition, respectively. We assume that the probability of damage progression in each damageable zone is known, fixed, and conditionally independent from the operational conditions. The transition probability associated with the DN action assumes that damage may develop in any subdomain Ω_m, with $m = 1, \ldots, N_\ell$, with a probability of 0.05, and then progress to the next δ interval with the same probability. Therefore, the DN *transition matrix* is a lower-left triangular matrix, with the highest probability assigned to remaining in the same state, consistent with the expected slow evolution of damage patterns typically observed in civil structures*. The transition to the next δ interval is the second most likely transition, while improvements have a zero probability. Once the structure reaches the last δ interval, it remains in this condition with a probability of 1. In contrast, the transition model assumed for the PM action maps the belief D_{t-1} to a belief D_t associated with a damage-free condition, independently of the condition at time $t-1$. The related transition matrix is an upper-right triangular matrix with probabilities all equal to 1 in the first row and 0 elsewhere.

*We recognize that the proper calibration of a transition matrix should account for the physical time duration between consecutive time steps — an aspect that has not been considered in the present investigation.

At each time step, the DT must suggest a control input $u_t^A \in \{\text{DN}, \text{PM}\}$. Opting for a DN action results in a positive reward but also carries the risk of deteriorating the structural health. Conversely, the PM action responds to the declining structural health but yields a negative reward. The computation of the costs associated with the health state and the control inputs is encapsulated by the ϕ_t^{reward} factor.

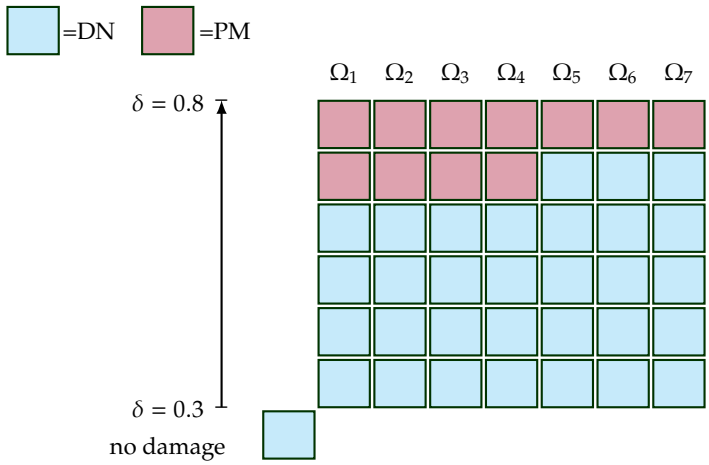


Figure 8.6: L-shaped cantilever beam - Deterministic control policy with two possible actions: DN (do nothing), and PM (perfect maintenance).

The two reward functions in Eq. (8.9) are chosen as follows:

$$R_t^{\text{control}}(u_t^A) = \begin{cases} +12, & \text{if } u_t^A = \text{DN}, \\ -20, & \text{if } u_t^A = \text{PM}, \end{cases} \tag{8.17}$$

$$R_t^{\text{health}}(d_t) = \begin{cases} +0.1, & \text{if } \mathscr{C} = 0, \\ -\exp(6\delta/5), & \text{if } \mathscr{C} = \{1,2,3,4\}, \\ -\exp(\delta), & \text{if } \mathscr{C} = \{5,6,7\}. \end{cases} \tag{8.18}$$

These non-dimensional rewards represent *indicative values* that the decision-maker is charged due to the condition of the asset. Although these values are not based on real data, *actual values* are typically available from lists of services and costs provided by state agencies and companies [319]. The three cases in R_t^{health} differentiate between the absence of damage, damage near the clamp, and damage far from the clamp, respectively. Note how they reflect the progressive deterioration of the structural health as a function of δ.

319. Papakonstantinou *et al.*, *Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation.* 2014

Offline, we solve the optimization problem in Eq. (8.8) to compute the control policy $\pi(D_t)$ — that is, the ϕ_t^{control} factor. The optimization of $R_t(U_t, D_t)$ is carried out considering a discount factor of $\gamma = 0.95$ and a weighting factor of $\chi_R = 2$. Fig. 8.6 illustrates the resulting control policy, which recommends that the asset operates until $\delta \in [65\%, 75\%]$ and $\delta \in [75\%, 80\%]$, respectively if $\mathscr{C} = \{1,2,3,4\}$ and if $\mathscr{C} = \{5,6,7\}$, at which point it should be repaired.

Fig. 8.7 depicts a *simulated online phase* of the DT up to time step $t_c = 50$. Results are reported in terms of the (unknown) ground truth digital state, and the corresponding DT estimate after assimilating the observational data. The graphs show the evolution of the digital state only for the damaged regions, nevertheless, damage can potentially affect all predefined damaged

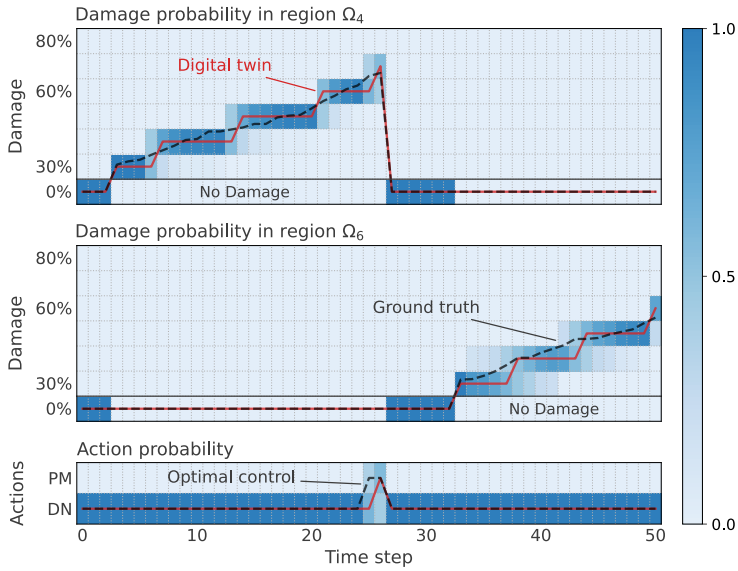


Figure 8.7: L-shaped cantilever beam - Online phase of the digital twin framework with two possible actions: DN (do nothing), and PM (perfect maintenance). Probabilistic and best point estimates of: (top) digital state evolution, against the ground truth digital state; (bottom) control inputs informed by the digital twin, against the optimal control input under the ground truth. In the top panels, the background color corresponds to $p(D_t|D_{t-1}, U^A_{t-1} = u^A_{t-1}, D^{\mathrm{NN}}_t)$. In the bottom panel, it corresponds to $p(U_t|D_t)$.

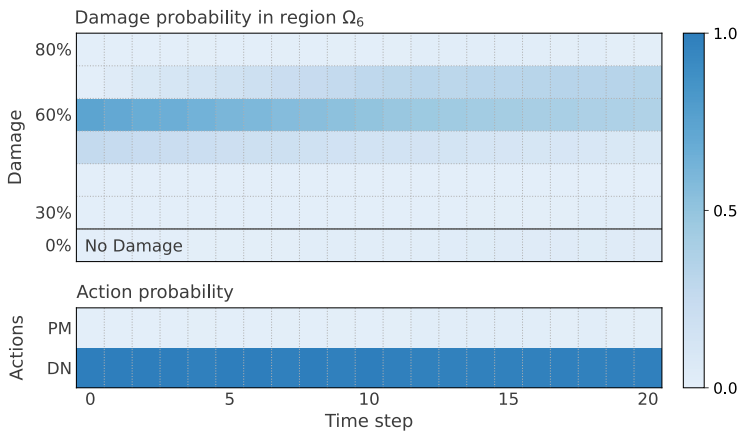


Figure 8.8: L-shaped cantilever beam - Digital twin future predictions with two possible actions: DN (do nothing), and PM (perfect maintenance). The starting time is $t_c = 50$. In the top panel, the probability $p(D_t|D_{t-1}, U_{t-1})$ relates to the amount of damage in Ω_6. In the bottom panel, it corresponds to $p(U_t|D_t)$.

regions $\Omega_1, \ldots, \Omega_7$. The DT proves capable of accurately tracking the digital state evolution with relatively low uncertainty. The corresponding estimation of the control inputs is reported in the bottom part of the figure, demonstrating that the DT promptly suggests the PM action within one time step of when the (unknown) ground truth structural health demands it.

Fig. 8.8 depicts the *predicted evolution* of the digital state and the corresponding control inputs, starting from $t_c = 50$. The prediction horizon is extended over 20 time steps into the future, so that $t_p = t_c + 20$. The DT prediction engine informs about the expected future degradation of structural health, allowing to plan preventive interventions and maintenance.

8.3.4 RESULTS: FOUR AVAILABLE ACTIONS

We now consider all four possible control inputs. We prescribe a stochastic degradation process with a probability of damage inception ($\mathscr{E} \neq 0$) equal to 0.5. Damage may develop in any of the predefined regions with $\delta \in [30\%, 70\%]$, and then propagate as in the previous case. This more aggressive degradation process is employed to fully display the decision-making capabilities of the DT in a short time frame. The effects of the MI and MA actions on the asset are simulated according to *stochastic repair processes*, where the structural health is forced to monotonically improve. For the MI action, we simulate δ decrements sampled from a Gaussian PDF centered at -12.5% with a standard deviation equal to 1%, while the MA action is modeled with δ decrements sampled from a Gaussian PDF centered at -17.5% with a standard deviation equal to 1%. For both actions, the damage-free condition is assumed to be restored if the resulting structural state features $\delta < 30\%$.

The transition model $p(D_t|D_{t-1}, U^A_{t-1} = u^A_{t-1})$ associated with the MI action assumes no improvements in the structural health with probability 0.1, improvements of one δ interval with probability 0.75, and improvements of two δ intervals with probability 0.15. The resulting CPT forms an upper-right triangular matrix, as deterioration from any state upon a repair action is assumed to have zero probability. The highest probability is assigned to improvements of one δ interval, followed by improvements of two δ intervals. The lower probability of remaining in the same deteriorated state instead reflects a failed maintenance. Similarly, the MA action assumes no improvements with probability 0.05, improvements of one δ interval with probability 0.3, improvements of two δ intervals with probability 0.4, and improvements of three δ intervals with probability 0.25. The related CPT is also an upper-right triangular matrix, where the highest probability is assigned to improvements of two δ intervals, followed by improvements of one δ interval, three δ intervals, and the lowest probability is associated with the possibility of a failed maintenance. Finally, the transition matrix associated with the DN action is modified with respect to the two-actions case. This modification involves spreading the probability of developing damage in any subdomain, which is 0.05, over the six δ intervals to accommodate the more aggressive degradation process.

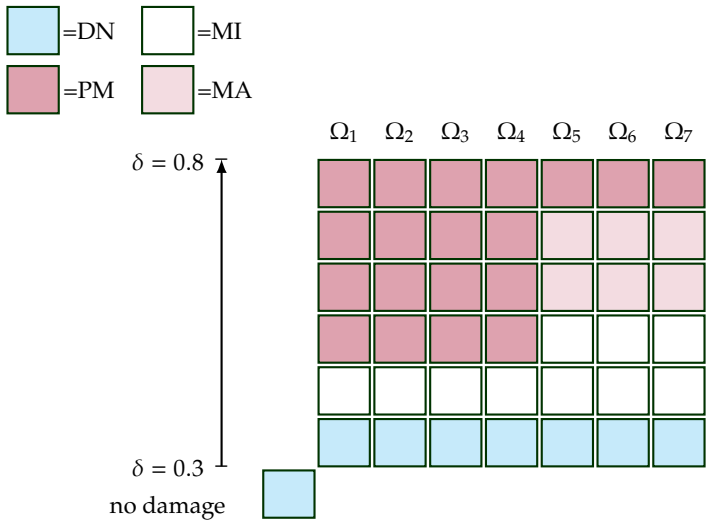


Figure 8.9: L-shaped cantilever beam - Deterministic control policy with four possible actions: DN (do nothing), PM (perfect maintenance), MI (minor imperfect maintenance) and MA (major imperfect maintenance).

The two reward functions in Eq. (8.9) are chosen as:

$$R_t^{\text{control}}(u_t^A) = \begin{cases} +12, & \text{if } u_t^A = \text{DN}, \\ -20, & \text{if } u_t^A = \text{PM}, \\ -8, & \text{if } u_t^A = \text{MI}, \\ -15, & \text{if } u_t^A = \text{MA}, \end{cases} \tag{8.19}$$

$$R_t^{\text{health}}(d_t) = \begin{cases} +0.1, & \text{if } \mathscr{d} = 0, \\ -\exp(5\delta), & \text{if } \mathscr{d} = \{1,2,3,4\}, \\ -\exp(4\delta), & \text{if } \mathscr{d} = \{5,6,7\}. \end{cases} \tag{8.20}$$

We consider a discount factor of $\gamma = 0.95$ and a weighting factor of $\chi_R = 2.5$. The resulting control policy (see Fig. 8.9) recommends that the asset should operate until $\delta \in [30\%, 35\%]$. Afterward, the recommended actions are as follows: if $\mathscr{d} = \{1,2,3,4\}$, a MI action should be performed when $\delta \in [35\%, 45\%]$, and a PM action should be performed when $\delta > 45\%$; if $\mathscr{d} = \{5,6,7\}$, a MI action should be performed when $\delta \in [35\%, 55\%]$, a MA action when $\delta \in [55\%, 75\%]$, and a PM action when $\delta > 75\%$.

Fig. 8.10 illustrates a simulated online phase of the DT up to $t_c = 50$. The DT accurately tracks the digital state evolution and timely suggests the appropriate control inputs most of the time. However, there are two instances, at time steps $t = 43$ and $t = 50$, where a sub-optimal action is proposed. In both cases, a MI action is proposed in place of a DN, as the DT estimates a $\delta \in [35\%, 45\%]$ rather than a $\delta \in [30\%, 35\%]$, related to a damage within Ω_7. This estimation error aligns with the confusion matrix in Fig. 8.5, highlighting the limited sensitivity of recordings to damage scenarios affecting the terminal region of the beam. While this particular misclassification is the most pathological one, and could potentially spoil the DT data assim-

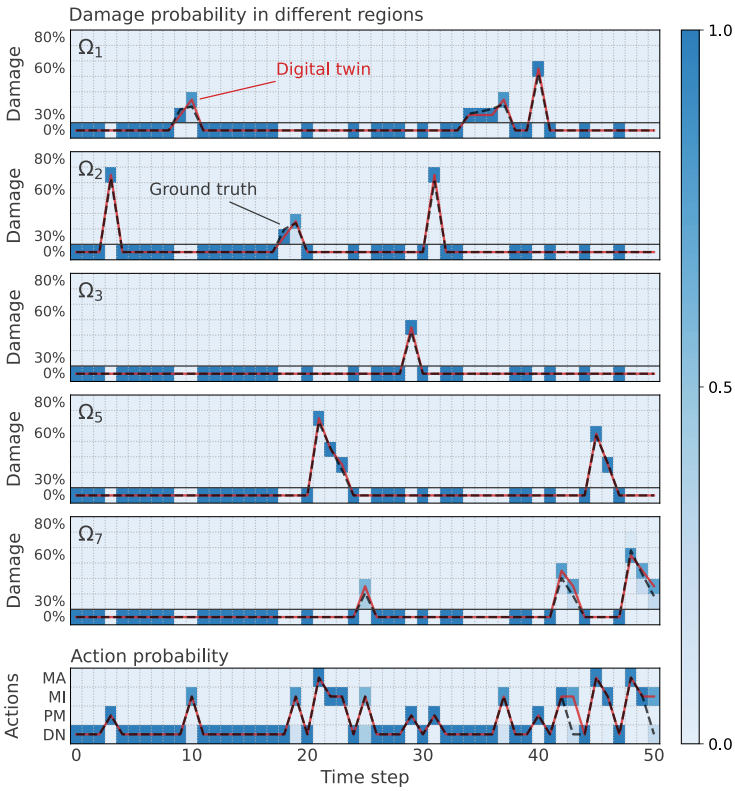


Figure 8.10: L-shaped cantilever beam - Online phase of the digital twin framework with four possible actions: DN (do nothing), PM (perfect maintenance), MI (minor imperfect maintenance), and MA (major imperfect maintenance). Probabilistic and best point estimates of: (top) digital state evolution, against the ground truth digital state; (bottom) control inputs informed by the digital twin, against the optimal control input under the ground truth. In the top panels, the background color corresponds to $p(D_t|D_{t-1}, U^A_{t-1} = u^A_{t-1}, D^{\mathrm{NN}}_t)$. In the bottom panel, it corresponds to $p(U_t|D_t)$.

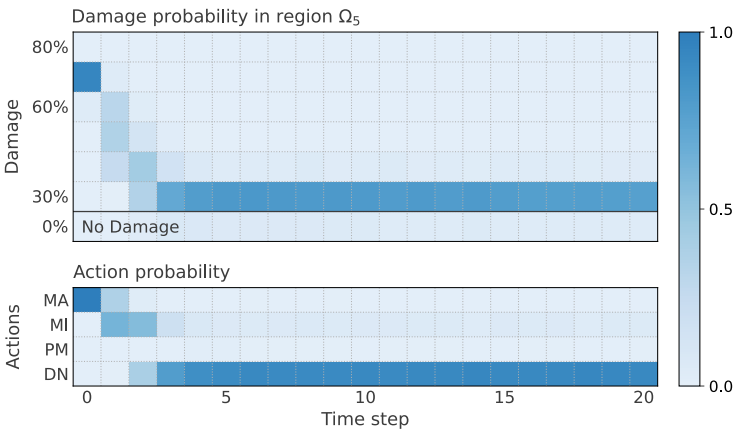


Figure 8.11: L-shaped cantilever beam - Digital twin future predictions with four possible actions: DN (do nothing), PM (perfect maintenance), MI (minor imperfect maintenance), and MA (major imperfect maintenance). The starting time is $t_c = 21$. In the top panel, the probability $p(D_t|D_{t-1}, U_{t-1})$ relates to the amount of damage in Ω_5. In the bottom panel, it corresponds to $p(U_t|D_t)$.

ilation capabilities, the DT promptly reverts to accurately track the structural health of the asset within one time step.

Fig. 8.11 illustrates the predicted evolution of the digital state and control inputs, from $t_c = 21$ over the next 20 time steps. The DT prediction correctly suggests a MA action with high probability, followed by two MI actions. Comparing the DT prediction with the actual evolution of the digital state and control inputs (refer to Fig. 8.10), we can see how closely the DT prediction resembles the observed evolution. Given that

the DT is not aware of the future values of the structural health parameters and the relative transition models do not match their stochastic evolution, this is a remarkable result demonstrating the prediction capabilities of the DT.

8.4 DEMONSTRATION: RAILWAY BRIDGE

This second case study involves the railway bridge previously examined in Sec. 5.4, Sec. 6.2.2, and Sec. 7.2.3.

The relevant SHM setup is as illustrated in Fig. 8.12, with synthetic displacement time histories $\mathbf{U}$ obtained from $N_u = 10$ sensors over a time interval $(0, T = 1.5\text{ s})$ with an acquisition frequency of $f_s = 400$ Hz. The recordings are then corrupted with an additive Gaussian noise to yield an SNR of 120.

The bridge experiences the passage of trains, traveling at a speed of $v \in [160, 215]$ km/h and carrying an axle mass of $m_A \in [16, 22]$ ton. In addition to the undamaged case, we consider $N_\ell = 6$ predefined damageable subdomains Ω_m, with $m = 1, \dots, N_\ell$, as depicted in Fig. 8.12. In these regions, the material stiffness can be reduced with a magnitude $\delta \in [30\%, 80\%]$, fixed over $(0, T)$.

8.4.1 DATASET ASSEMBLY

Apart from the damage scenarios, the FOM is identical to that used in Sec. 7.2.3, featuring $N_{\text{FE}} = 17,292$ DOFs. The ROM is derived from a snapshot matrix $\mathbf{S}$, assembled through $N_P = 400$ evaluations of the FOM for different values of $\boldsymbol{\mu} = (v, m_A, \ell, \delta)^\top$. Setting the error tolerance in Eq. (3.16) to $\epsilon_\sigma = 10^{-3}$, we consider $N_{\text{RB}} = 133$ POD modes. By taking $\boldsymbol{\mu}$ as uniformly distributed and sampling it via Latin hypercube rule, $\mathcal{D}_{\text{DT}}$ is built with $N_{\text{train}} = 10,000$ instances for training NN_{CL} and NN_{RG}.

8.4.2 DIGITAL TWIN FRAMEWORK

The testing phase of NN_{CL} and NN_{RG} is carried out using noisy FOM solutions, while the DT online phase is simulated by assimilating $N_{\text{obs}} = 1$ observations $\mathbf{U}^{\text{EXP}}$, enabling the DT to estimate the variation in the structural health parameters each time a train traverses the bridge. The value range of δ is discretized into $N_\delta = 6$ intervals, consistent with those considered for the L-shaped beam. This results into $N_{\mathcal{D}} = 37$ possible digital states, sorted first by damage location and then by damage level.

The confusion matrix measuring the offline performance of NN_{CL} and NN_{RG} is reported in Fig. 8.13, computed following

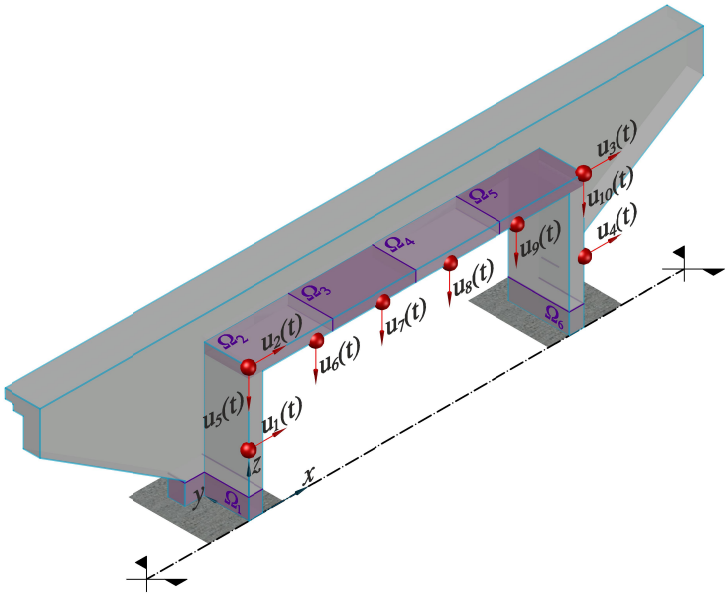


Figure 8.12: Railway bridge: details of synthetic recordings related to displacements $u_1(t), \ldots, u_{10}(t)$, and predefined damage regions $\Omega_1, \ldots, \Omega_6$.

the same setup used for the L-shaped beam. The actual digital state is detected with a classification accuracy of 91.39%. Here, most misclassifications occur due to confusing adjacent digital states within the same damage location, resulting in a tridiagonal band matrix.

We consider the following three possible control inputs:

I *Do nothing* (DN) action. No maintenance action is performed, and the physical state evolves following a stochastic deterioration process.

II *Perfect maintenance* (PM) action. A maintenance action is performed, restoring the asset from its current condition to the damage-free state.

III *Restrict operational conditions* (RE) action. The operational conditions of the bridge are restricted by allowing only lightweight trains, carrying less than 18 ton per axle, to travel across the bridge. Such a restriction results in a lower deterioration rate, but also yields a lower revenue generated by the infrastructure, as described below.

In cases where the most recently issued control input is either DN or RE, the physical state undergoes a degradation process that monotonically deteriorates the structural health. When operational conditions are not restricted, we prescribe a stochastic degradation process featuring a probability of damage inception ($\ell \neq 0$) equal to 0.5. Damage may develop in any of the N_ℓ predefined regions with $\delta \in [30\%, 35\%]$, and then propagate

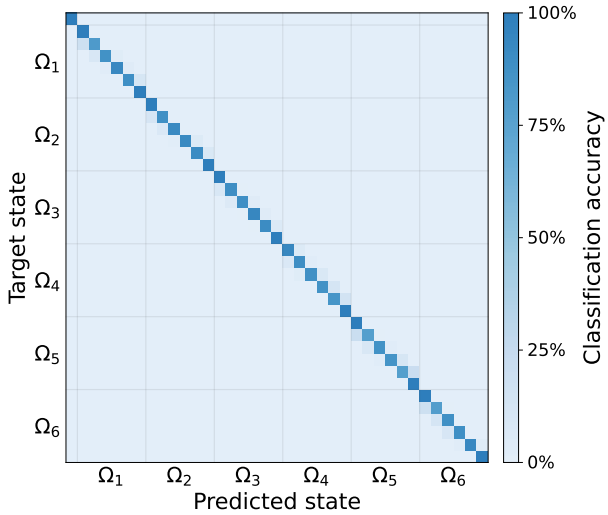


Figure 8.13: Railway bridge - Confusion matrix measuring the offline performance of the DL models in correctly categorizing the digital state. Results are reported in terms of classification accuracy, measuring how observational data are classified with respect to the ground truth digital state. Digital states are ordered first for damage location and then for damage level.

with δ increments sampled from a Gaussian PDF centered at 1.5% with a standard deviation of 1%. When operations are restricted and only lightweight trains are allowed to travel across the bridge, we instead assume a probability of damage inception of 0.25. In this case, damage may develop with $\delta \in [30\%, 35\%]$, and then propagate with δ increments sampled from a Gaussian PDF centered at 0.95% with a standard deviation of 0.5%. In both cases, negative increments are rounded to zero. The resulting trajectory of the structural health parameters is intended to represent periods of gradual degradation in the structural health, as well as sudden changes due to discrete damage events. Also in this case, the effect of a PM action is simulated by restoring the physical state to its undamaged configuration.

The transition model associated with the DN action assumes that damage may start in any subdomain Ω_m, with $m = 1, \ldots, N_\ell$, with a probability of 0.1, and then progress to the next δ interval with the same probability. For the RE action, this probability is assumed to decrease to 0.03, removing the assumption that the probability of damage progression is conditionally independent from the operational conditions. Consequently, the CPTs associated with the DN and RE actions are lower-left triangular transition matrices. The highest probability is assigned to remaining in the same state, followed by the transition to the next δ interval, while there is zero probability of improvement. In contrast, the transition model assumed for the PM action maps D_{t-1} to a belief D_t associated with a damage-free condition, regardless of the condition at time $t-1$.

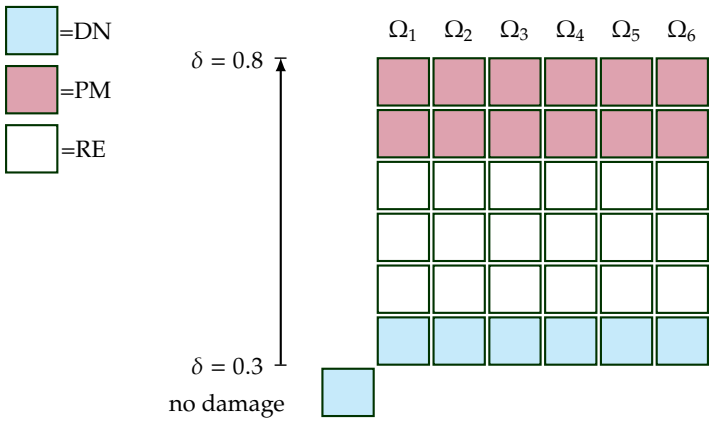


Figure 8.14: Railway bridge - Deterministic control policy: DN (do nothing), PM (perfect maintenance), and RE (restrict operational conditions).

The two reward functions in Eq. (8.9) are defined as follows:

$$R_t^{\text{control}}(u_t^A) = \begin{cases} +30, & \text{if } u_t^A = \text{DN}, \\ -250, & \text{if } u_t^A = \text{PM}, \\ +27, & \text{if } u_t^A = \text{RE}, \end{cases} \tag{8.21}$$

$$R_t^{\text{health}}(d_t) = \begin{cases} +0, & \text{if } \mathscr{E} = 0, \\ -\exp(5\delta) + 4, & \text{if } \mathscr{E} \neq 0, \\ -250, & \text{if } \delta \geq 79\%, \end{cases} \tag{8.22}$$

where the last contribution in R_t^{health} penalizes excessively compromised structural states with a significantly negative reward.

Offline, we solve the optimization problem in Eq. (8.8) by considering a discount factor of $\gamma = 0.90$ and a weighting factor of $\chi_R = 1$. The resulting control policy (see Fig. 8.14) recommends that the asset operates under normal conditions until the damage level reaches $\delta \in [30\%, 35\%]$. Beyond this point, it advises to switch to the more conservative RE to minimize further degradation. Once the damage level reaches $\delta \geq 65\%$, the bridge should be repaired.

8.4.3 RESULTS

Fig. 8.15 presents a sample simulation of the DT online phase up to time step $t_c = 60$. Initially, damage develops within Ω_5, and the DT follows its evolution with a limited delay of at most two time steps compared to the ground truth, attributed to the need to update the relative prior belief from previous time steps. The RE action is suggested as soon as the DT estimates a $\delta \in [35\%, 65\%]$, after which point it continues tracking the structural health parameters that evolve with a lower deterioration rate. Finally, a PM action is suggested due to an excessively compromised structural state. A similar behavior is observed for the subsequent damage scenario affecting Ω_6.

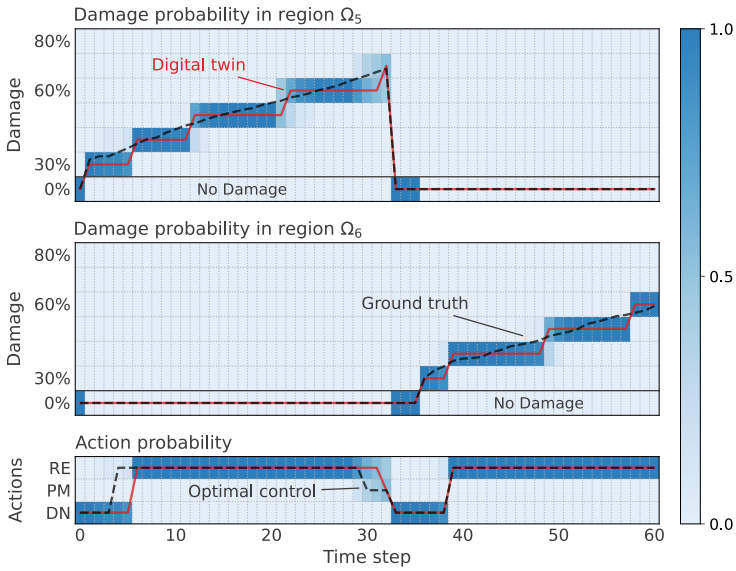


Figure 8.15: Railway bridge - Online phase of the digital twin framework with three possible actions: DN (do nothing), PM (perfect maintenance), and RE (restrict operational conditions). Probabilistic and best point estimates of: (top) digital state evolution, against the ground truth digital state; (bottom) control inputs informed by the digital twin, against the optimal control input under the ground truth. In the top panels, the background color corresponds to $p(D_t|D_{t-1}, U^A_{t-1} = u^A_{t-1}, D^{\mathrm{NN}}_t)$. In the bottom panel, it corresponds to $p(U_t|D_t)$.

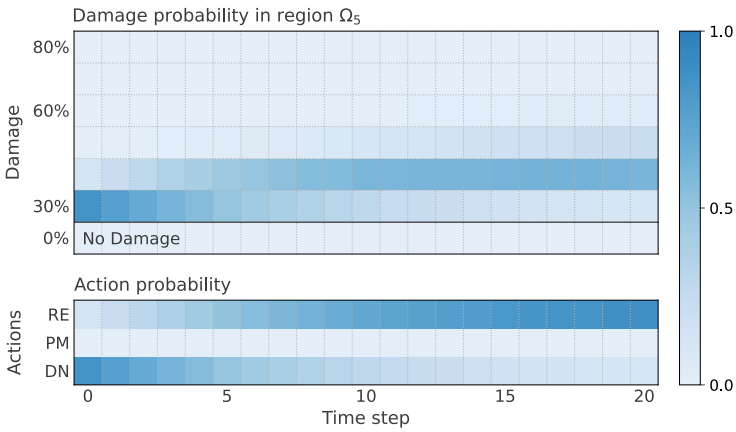


Figure 8.16: Railway bridge - Digital twin future predictions with three possible actions: DN (do nothing), PM (perfect maintenance), and RE (restrict operational conditions). The starting time is $t_c = 5$. In the top panel, the probability $p(D_t|D_{t-1}, U_{t-1})$ relates to the amount of damage in Ω_5. In the bottom panel, it corresponds to $p(U_t|D_t)$.

Fig. 8.16 presents the predicted evolution of the digital state and control inputs, starting from $t_c = 5$, just before taking the RE action, and spanning over the next 20 time steps. The DT initially forecasts the expected degradation of the structural health based on the transition model associated with the DN action, followed by a relatively high probability prediction of taking a RE action after a few time steps. While the DT prediction closely aligns with the actual evolution of the digital state and control inputs observed during the online phase (see Fig. 8.15), we can spot a couple of discrepancies. First, the estimated digital state lags behind the ground truth value by two time steps. Second, the prediction displays an overly optimistic deterioration rate, indicating the potential need for a more refined transition model.

8.5 CONCLUSIONS

In this Chapter, we have proposed a predictive DT approach for the dynamic health monitoring and management of engineering structures. The framework relies on a PGM inspired by [19], which encodes the asset-twin dynamical system, the end-to-end information flow via observational data and control inputs, and its evolution over time, all with quantified uncertainty.

19. Kapteyn *et al.*, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*. 2021

Observational data are assimilated using DL models to automatically extract damage-sensitive features and assess the structural state in real time. This process continually updates a digital state comprising structural health parameters through sequential Bayesian inference. The updated digital state is then exploited to forecast its future evolution, enabling predictive decision-making about maintenance and management actions. A preliminary offline phase involves: (i) using ROMs to generate datasets for training the DL models; (ii) learning a control policy to map the belief over the digital state onto control inputs.

The DT framework enables a bi-directional information flow between the monitored asset and its virtual replica. Through this closed information loop, we have moved beyond the digital model and digital shadow instances discussed in the previous Chapters. In particular, by integrating data assimilation capabilities with an automated control interaction, we have demonstrated the realization of value within an SHM context.

The proposed strategy has shown capable to track the structural health evolution and promptly suggest the most appropriate control input with relatively low uncertainty. Its effectiveness has been assessed through the simulated monitoring of an L-shaped cantilever beam and a railway bridge, showcasing the dynamic decision-making capabilities of health-aware DTs. The suggested actions, which encompass interventions directly affecting the physical state or influencing the operational conditions, have been suggested within at most two time steps of when the ground truth structural health demands them. When the DT has failed to match the ground truth condition, it has shown capable to correct itself after assimilating new observational data.

While the capabilities of health-aware DTs have been demonstrated in the specific context of civil structures, the applicability of our framework is general. The graph topology can be reorganized to accommodate situations where observational data are not acquired after issuing a control input, or when control inputs are issued with a different frequency than that governing the DT update. The solution to the inverse problem (if any) can be estimated using methods other than DNNs, such as MCMC

sampling. The state transition models are closely tied to the chosen parametrization of the digital state. Similarly, the control actions may vary depending on the context.

Regarding the limitations, the primary concern is again related to the modeling assumptions and their impact on the reliability of the digital state identification. Assuming that the computational model has been previously calibrated to reflect the characteristics of the asset and minimize the model inadequacy, the identification of damage patterns relies solely on modeling assumptions. However, the flexibility of our PGM allows us to envision two possible ways for mitigating the impact of mis-specified damage descriptions. First, the introduction of quantities of interest to perform posterior predictive checks on the tracking capabilities of the DT would be useful to assess the digital state estimate and, if needed, predispose in situ inspections. Second, there is the possibility of using control-dependent and digital state-dependent data from previous time steps as additional observations for conditioning the structural health identification.

As suggested by the railway bridge case study, more accurate transition models would enable predictive decision-making that is better tailored to the monitored asset. While physics-based models can be used to derive state transition probabilities, future research lines should also explore the possibility to learn these probabilities from online data. For instance, transition probabilities can be modelled as random variables with known distributions, such as conjugate distributions. This would allow for capturing the stochastic nature of state transitions and refining our belief after acquiring new sensor data. The result would be a more personalized and robust DT, capable of accommodating non-stationary evolutions of the physical state. Additionally, not only we would be able to refine the control policy at each time step, but we could also compute risk measures to account for rare events. Different policies could be computed for various risk measures, allowing for an online policy switch.

Currently, the control policy is computed under the simplifying assumptions of a fully observable state and an infinite planning horizon. However, this precludes accounting for the partial observability of the state and incorporating information-gathering actions, such as installing new sensors or predisposing inspections. Moreover, considering a finite planning horizon representing the design lifetime of the asset would be a more realistic alternative. Finally, only discrete random variables are currently supported by our framework, which should be extended to support continuous random variables as well.

9 CONCLUSIONS AND OUTLOOK

This Thesis has aimed to advance computational techniques for enabling predictive digital twins of structural systems. The proposed contributions toward this goal have leveraged state-of-the-art methodologies from data science and engineering, while also introducing novel perspectives in model-based and data-driven approaches to structural health monitoring. In this Chapter, we outline the main contributions of this Thesis, discuss their limitations, and envision possible future developments.

9.1 SUMMARY OF CONTRIBUTIONS

The application of the digital twin paradigm in structural health monitoring must meet several requirements. According to the 2023 report by the *National Academies of Engineering, Science, and Medicine* [10], a digital twin can be distinguished from simpler instances of digital models or digital shadows:

> "The key elements that comprise a digital twin include (*i*) modeling and simulation to create a virtual representation of a physical counterpart, and (*ii*) a bi-directional interaction between the virtual and the physical. This bi-directional interaction forms a feedback loop that comprises dynamic data-driven model updating and optimal decision-making."

10. National Academies of Sciences, Engineering, and Medicine, *Foundational Research Gaps and Future Directions for Digital Twins*. 2023

Once again, we stress how the digital twin paradigm aims at integrating data assimilation capabilities with automated control interactions to enable a bi-directional information flow between the monitored asset and its virtual replica.

Building upon the challenges and opportunities outlined in Chapter 1, we have formulated the objectives through which this Thesis aims to contribute. These contributions share the same fundamental aim of advancing toward reliable, possibly real-time, early identification of damage patterns in civil structures. This is crucial for dynamically updating computational models and integrating them into a data assimilation and feedback-and-control loop throughout the asset life cycle. This challenge has been addressed through the combined use of simulation-based supervision, to overcome the unavailability of labeled knowledge for damaged structures, and learnable mappings to

handle and process high-dimensional data, enhance damage identification capabilities, and improve computational efficiency.

Firstly, we have provided a theoretical foundation and cutting-edge insights relevant to the areas explored in this Thesis. In particular, we have highlighted the role of parameter identification from sensor measurements as a critical element for the development of digital twins of engineering structures. Therefore, in Chapter 2 we have outlined a methodological distinction between model-based and data-driven approaches to parameter identification — both of which offer viable solutions to address the problem of structural health monitoring. Subsequently, in Chapter 3 we have focused on computational models within a structural dynamics setting. We have shown how these models can replicate structural responses under varying operational and damage conditions, enabling the development of damage identification strategies within the simulation-based paradigm of structural health monitoring. Additionally, we have introduced reduced-order modeling based on a reduced basis method to tackle many-query tasks involving the repeated evaluation of computationally intensive forward models. To conclude our critical review, in Chapter 4 we have analyzed various machine learning methods, envisioned to support physics-based modeling in realizing effective digital twins for structural health monitoring. Here, the aim of extracting information from observations, predicting future system responses, and determining optimal courses of action, has been addressed introducing regression and classification perspectives, deep learning architectures, and different types of Bayesian networks. Throughout this Thesis, these methods have been employed to support three main fronts: (i) surrogate modeling to emulate input-output maps; (ii) data assimilation and inverse problems related to the identification of structural health parameters from sensor measurements; and (iii) reasoning and decision-making under uncertainty to enable automatic interactions in asset-twin systems.

Subsequently, the achieved objectives have been identified as four main contributions, each presented in a different Chapter. These contributions aim not only to address specific challenges in the literature, but also to trace an incremental path towards increasingly comprehensive digital twin technologies.

In Chapter 5, we have developed a data-driven damage localization framework by leveraging deep metric learning and simulation-based supervision. This Chapter has focused on learning a mapping of raw vibration recordings onto a low-dimensional space where structural health parameters can be easily discriminated. The use of pairwise training within a

Siamese architecture has demonstrated remarkable feature extraction capabilities, as well as robustness against variations in the extent and severity of damage. However, despite achieving an impressive identification accuracy, this framework does not account for any sources of uncertainty. Therefore, in Chapter 6, we have shifted to model-based approaches. Here, we have addressed the computational efficiency and scalability of surrogate models for structural health monitoring through multi-fidelity deep neural networks. This Chapter has focused on using full and reduced-order numerical models to sequentially train surrogate components with increasing levels of approximation. This strategy has enabled superior accuracy compared to single-fidelity approximations without increasing computational costs. The achieved surrogates have proven useful both for model-updating purposes and for generating large training datasets inexpensively. However, updating structural health parameters by directly using raw measures may face limitations due to the slow convergence of sampling algorithms and the curse of dimensionality affecting the inference of low-sensitivity damage parameters. In Chapter 7, we have therefore proposed a hybrid data/model approach for real-time damage identification, capitalizing on the strengths of both worlds. This Chapter has focused on enhancing Bayesian model updating through learnable mappings. Two data-driven models — one to extract features from raw sensor recordings, and the other to emulate the relationship between parameters and features — have been synergistically used within a Markov chain Monte Carlo algorithm to sample the posterior distribution of parameters of interest. We have exploited deep metric learning for feature extraction, enabling a damage identification accuracy comparable to that achieved in Chapter 5, and multi fidelity surrogate modeling for generating large training datasets. Simultaneously, the sampling algorithm has maintained a measure of the reliability of the estimates provided, with improved computational efficiency due to the low-dimensionality of the involved features.

Finally, in Chapter 8 we have developed a digital twin concept for predictive decision-making, informed by structural health diagnostics. The asset-twin interaction has been framed within a probabilistic graphical model, which encodes the end-to-end information flow via observational data and control inputs. We have integrated deep learning data assimilation capabilities within a sequential Bayesian inference formulation to update the digital twin. Additionally, by incorporating automated control interactions in the form of a Markov decision process, we have advanced beyond the digital model and digital shadow instances

achieved in previous contributions. The health-aware digital twins have demonstrated their ability to track the evolution of structural health parameters, forecast future trajectories of structural health, and inform predictive decision-making.

9.2 LIMITATIONS

Throughout this dissertation, we have highlighted some concerns regarding the proposed methodologies. Here, our aim is to gather and discuss these concerns.

Methodological Perspective. An important aspect to highlight is that all contributions of this Thesis have been made purely from a methodological perspective. Technological and implementation aspects, such as how the structural response is experimentally perceived and transferred for downstream processing, have not been explicitly addressed. Similarly, validating the proposed methodologies against suitable experimental settings also appears necessary.

Modeling Assumptions. A second concern pertains to the uncertainties associated with modeling assumptions. Throughout this Thesis, the simulation-based paradigm of structural health monitoring has been employed to describe the effect of damage on the structural response, enabling damage identification capabilities beyond simple damage detection. However, even with a computational model calibrated to reflect the unique characteristics of the monitored structure, the identification of damage patterns relies solely on modeling assumptions, which may suffer from representability issues. Specifically, damage has been systematically described as a localized reduction of material stiffness, rather than directly modeling the specific damage pattern. This approach allows for a time scale separation between damage growth and damage assessment [43], while treating the structure as a linear time-invariant system within an observation interval*. Although characterizing the specific type of damage that has emerged can be addressed subsequent to early identification through a detailed in situ inspection, not all damage patterns can be described in this way. We can not claim to identify damage scenarios that do not lead to an observable loss of stiffness within localized areas of the structure. However, an inaccurate damage description could pose a risk to the reliability of a parameter identification strategy upon deployment. Therefore, addressing how to properly account for model-form uncertainty in this respect requires a dedicated investigation.

43. Eftekhar Azam *et al.*, *Online damage detection in structural systems via dynamic inverse analysis: A recursive Bayesian approach*. 2018

*Given the short duration of the observation window compared to the time necessary for long-term degradation processes to advance, this modeling assumption is generally accepted for civil structures.

Parametrization Choice. A related aspect that directly affects the performance of a damage identification strategy is the selection of an appropriate parametrization. A simple and intuitive approach could involve sensitivity analysis, which enables selecting the parameters that have the most significant influence on the model output. However, it is not guaranteed that regions or structural elements associated with high sensitivity are also linked to damage mechanisms of interest. Therefore, the choice of an appropriate parametrization should be carefully addressed to allow for a balanced compromise between describing damage mechanisms of interest and ensuring identifiability. Additionally, issues of identifiability and observability in dynamic systems have not been addressed, and further considerations in these respects are also necessary.

Fixed Parametrization. In our digital twin framework, the damage parameterization directly shapes the digital state space, which in turn determines the expressiveness of the digital twin. Since the capability of the digital state space to represent the state of the physical asset impacts its decision-making capabilities, an effective digital twin should accurately capture the unknown physical state and its evolution. Currently, our digital twin framework only supports a fixed digital state space. However, it should instead incorporate adaptive capabilities to better accommodate the challenge of designing a sufficiently expressive digital state space a-priori — an aspect that would enable the possibility of refinement and enrichment in case of inaccuracies.

Limited Twinning Capabilities. In relation to our digital twin framework, another aspect to highlight is that it represents a starting point for a more comprehensive realization. Indeed, our framework stands out from simpler instances of digital models or digital shadows. However, numerous improvements are still necessary to maximize the value we can realize from the use of digital twins for structural health monitoring purposes. Without aiming to be exhaustive, relevant examples include: developing a suitable calibration platform to establish an asset-twin connection capable of accurately reflecting the unique characteristics of the monitored structure; introducing the concept of adaptivity in terms of digital state space and state transition models; enabling the computation of quantities of interest to check the tracking capabilities of the digital twin; accounting for relevant environmental information to address potentially confounding influences, similarly to [106], and incorporating knowledge about time-dependant deterioration processes due to aggressive

106. Torzoni *et al.*, *SHM under varying environmental conditions: An approach based on model order reduction and deep learning*. 2022

environments, see, e.g., [354]; integrating multi-scale twinning capabilities to provide a detailed component-level assessment of the structural reliability, see, e.g., [355]; considering a finite planning horizon representing the design lifetime of the asset; addressing the partial observability of the physical state and incorporating information-gathering actions; and extending the framework to accommodate continuous random variables.

354. Biondini *et al.*, *Probabilistic service life prediction and maintenance planning of deteriorating structures*. 2005

355. Biondini *et al.*, *Probabilistic Service Life Assessment and Maintenance Planning of Concrete Structures*. 2006

Overfitting of Training Data. A drawback shared by all machine learning algorithms we have used is their susceptibility to overfitting the training data, which may lead to overconfident predictions. This issue is closely related to the necessity of using potentially time-consuming high-fidelity numerical models to generate sufficiently informative training datasets. In this regard as well, a dedicated investigation on how to account for model-form uncertainty would be beneficial.

9.3 Opportunities for Further Research

In what follows, we finally discuss some possible avenues for future research directions that may build upon the various challenges addressed in this Thesis.

Optimal Sensing. One aspect of interest involves designing sensor systems to be as sensitive as possible to potential damage conditions affecting the monitored structure. Employing a sensor placement strategy — rather than solely relying on engineering intuition, as done throughout this Thesis — would be crucial for maximizing the effectiveness of information for damage assessment. This aspect not only pertains to pre-designing a sensor network but also holds potential for responding to the need to install new sensors. Ideally, we may be able to make decisions about where to install new sensors among a (potentially large) set of candidate locations. Available observational data could be combined with physics-based models to simulate how each sensor would respond to different asset states. These "what-if" scenarios would allow us to investigate the enrichment of the sensing network under sensor budget constraints while learning the corresponding data assimilation model to optimize the cost-versus-estimation performance trade-off.

Edge Computing and Co-Located Twins. Another attractive area of research involves leveraging sensors with onboard computing resources. Since trained deep learning models only require a sequence of affine transformations and nonlinear

activation functions, low-cost micro sensors with onboard computing capabilities could efficiently process observational data at the sensor level. This perspective offers clear advantages for decentralizing data processing, saving memory storage, optimizing energy consumption, and avoiding problems related to data transmission. Moreover, deploying smart sensing systems has the potential to further advance the digital twin paradigm, facilitating its ubiquitous adoption for creating networks of digital twins, thereby providing a new level of monitoring and decision support.

Online Learning. Previously, we have highlighted the current lack of adaptivity in our digital twin framework. Online learning offers an attractive opportunity to adaptively fine-tune a digital twin. Such adaptivity should target at least two aspects: the digital state space and the state transition models.

Given the challenge of pre-designing a digital state space expressive enough to capture all possible future trajectories of the physical state, the digital state space should allow for refinement and enrichment in case of inaccuracies. This aspect could be managed by initially computing quantities of interest to assess the tracking capabilities of the digital twin. If necessary, experimental design techniques, such as Bayesian experimental design [356], could then be employed to determine the optimal way to enrich the digital state space.

356. Rainforth *et al.*, *Modern Bayesian Experimental Design*. 2024

As discussed in Chapter 8, the digital twin should also be capable of adapting to rare events, such as atypical environmental conditions. To compensate for limited or unavailable experimental data, transition probabilities tailored to the monitored asset could initially be derived from physics-based models, and then fine-tuned on the online data stream. This could be conveniently achieved by modeling transition probabilities as random variables with conjugate distributions, enabling us to capture the stochastic nature of state transitions and refine our beliefs after acquiring new sensor data. This approach would allow us to overcome the unrealistic assumption that the dynamics of the underlying Markov decision process is precisely known. Additionally, not only we would be able to refine the control policy after every update, but we could also compute risk measures to account for different events. Different policies could be computed for various risk measures, allowing for an online policy switch in the case of rare events.

Model-Form Uncertainty. Further work is needed to address concerns related to modeling assumptions. Despite efforts

to refine the structural model or employ more sophisticated descriptions for potential damage patterns, engineering structures may be too complex and uncertain systems to be perfectly modeled. Uncertainties arising from modeling choices, environmental factors, and operational variabilities are among the many factors that can hinder the model capability to faithfully describe a real-world system. In terms of monitoring, properly accounting for model-form uncertainty is crucial to obtain a reliable solution to the parameter identification problem. To address this challenge, latent variable models offer an interesting perspective. Latent variables could be leveraged to account for missing pieces of information, and examining their distribution could provide insights into the reliability of the forward model and, consequently, of the monitoring process. Using latent variable models for data assimilation purposes would require the use of appropriate Bayesian inference engines. In this regard, the recent advances in simulation-based inference stand as an appealing choice [357].

357. Cranmer *et al.*, *The frontier of simulation-based inference*. 2020

Experimental Data. A last aspect that should be explored concerns the use of available experimental recordings to improve the reliability of data assimilation models. To bridge the data-to-methodology gap for supervised learning tasks, we envision the application of multi-fidelity methods at the feature level as an opportunity to advance current strategies in population-based [116,118] and domain adaptation [48,120,122] for structural health monitoring. For example, experimental data from existing structures could be integrated with synthetic data from physics-based models via multi-fidelity information fusion to establish shared feature spaces unaffected by label inconsistencies.

116. Bull *et al.*, *Foundations of population-based SHM, Part I: Homogeneous populations and forms*. 2021

118. Gardner *et al.*, *Foundations of population-based SHM, Part III: Heterogeneous populations – Mapping and transfer*. 2021

48. Giglioni *et al.*, *A domain adaptation approach to damage classification with an application to bridge monitoring*. 2024

120. Wang *et al.*, *Knowledge transfer for structural damage detection through re-weighted adversarial domain adaptation*. 2022

122. Poole *et al.*, *On statistic alignment for domain adaptation in structural health monitoring*. 2023

BIBLIOGRAPHY

1 Digital Twin: Definition & Value — An AIAA and AIA Position Paper, "AIAA Digital Engineering Integration Committee and others," AIAA: Reston, Virginia, 2020. cited on p. 2

2 Tuegel, E. J., Ingraffea, A. R., Eason, T. G., and Spottswood, S. M., "Reengineering Aircraft Structural Life Prediction Using a Digital Twin," *International Journal of Aerospace Engineering*, Vol. 2011, 2011, p. 154 798.
DOI: 10.1155/2011/154798 cited on p. 2

3 Tuegel, E., "The Airframe Digital Twin: Some Challenges to Realization," *53th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Honolulu, Hawaii, 2012.
DOI: 10.2514/6.2012-1812 cited on p. 2

4 Niederer, S. A., Sacks, M. S., Girolami, M., and Willcox, K., "Scaling digital twins from the artisanal to the industrial," *Nature Computational Science*, Vol. 1, No. 5, 2021, pp. 313–320.
DOI: 10.1038/s43588-021-00072-5 cited on p. 2

5 Moya, B., Alfaro, I., Gonzalez, D., Chinesta, F., and Cueto, E., "Physically sound, self-learning digital twins for sloshing fluids," *PLoS One*, Vol. 15, No. 6, 2020, e0234569.
DOI: 10.1371/journal.pone.0234569 cited on p. 2

6 Thelen, A., Zhang, X., Fink, O., Lu, Y., Ghosh, S., Youn, B. D., Todd, M. D., Mahadevan, S., Hu, C., and Hu, Z., "A comprehensive review of digital twin – part 1: Modeling and twinning enabling technologies," *Structural and Multidisciplinary Optimization*, Vol. 65, 2022, p. 354.
DOI: 10.1007/s00158-022-03425-4 cited on p. 2

7 Grieves, M. and Vickers, J., "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," *Transdisciplinary Perspectives on Complex Systems: New findings and approaches*, 2017, pp. 85–113.
DOI: 10.1007/978-3-319-38756-7_4 cited on p. 2

8 Chinesta, F., Cueto, E., Abisset-Chavanne, E., Duval, J. L., and Khaldi, F. E., "Virtual, Digital and Hybrid Twins: A New Paradigm in Data-Based Engineering and Engineered Data," *Archives of Computational Methods in Engineering*, Vol. 27, 2020, cited on pp. 2, 113

pp. 105–134.
DOI: 10.1007/s11831-018-9301-4

9 Rasheed, A., San, O., and Kvamsdal, T., "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, Vol. 8, 2020, pp. 21 980–22 012. cited on p. 2
DOI: 10.1109/ACCESS.2020.2970143

10 National Academies of Sciences, Engineering, and Medicine, *Foundational Research Gaps and Future Directions for Digital Twins*. Washington, DC: The National Academies Press, 2023. cited on pp. 2, 213
DOI: 10.17226/26894

11 Li, C., Mahadevan, S., Ling, Y., Choze, S., and Wang, L., "Dynamic Bayesian Network for Aircraft Wing Health Monitoring Digital Twin," *AIAA Journal*, Vol. 55, No. 3, 2017, pp. 930–941. cited on pp. 4, 84
DOI: 10.2514/1.J055201

12 Ye, Y., Yang, Q., Yang, F., Huo, Y., and Meng, S., "Digital twin for the structural health management of reusable spacecraft: A case study," *Engineering Fracture Mechanics*, Vol. 234, 2020, p. 107 076. cited on p. 4
DOI: 10.1016/j.engfracmech.2020.107076

13 Podskarbi, M. and Knezevic, D. J., "Digital Twin for Operations - Present Applications and Future Digital Thread," *OTC Offshore Technology Conference*, Houston, Texas, 2020. cited on p. 4
DOI: 10.4043/30553-MS

14 Phua, A., Davies, C., and Delaney, G., "A digital twin hierarchy for metal additive manufacturing," *Computers in Industry*, Vol. 140, 2022, p. 103 667. cited on p. 4
DOI: 10.1016/j.compind.2022.103667

15 Jans-Singh, M., Leeming, K., Choudhary, R., and Girolami, M., "Digital twin of an urban-integrated hydroponic farm," *Data-Centric Engineering*, Vol. 1, 2020, p. 20. cited on p. 4
DOI: 10.1017/dce.2020.21

16 Tzachor, A., Sabri, S., Richards, C. E., Rajabifard, A., and Acuto, M., "Potential and limitations of digital twins to achieve the Sustainable Development Goals," *Nature Sustainability*, Vol. 5, No. 10, 2022, pp. 822–829. cited on p. 4
DOI: 10.1038/s41893-022-00923-7

17 Arcieri, G., Hoelzl, C., Schwery, O., Straub, D., Papakonstantinou, K. G., and Chatzi, E., "Bridging POMDPs and Bayesian decision making for robust maintenance planning under model uncertainty: An application to railway systems," *Reliability Engineering & System Safety*, 2023, p. 109 496. cited on pp. 4, 84
DOI: 10.1016/j.ress.2023.109496

18 McClellan, A., Lorenzetti, J., Pavone, M., and Farhat, C., "A Physics-Based Digital Twin for Model Predictive Control of Autonomous Unmanned Aerial Vehicle Landing," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 380, No. 2229, 2022, p. 20 210 204. DOI: 10.1098/rsta.2021.020417 cited on p. 4

19 Kapteyn, M. G., Pretorius, J. V., and Willcox, K. E., "A probabilistic graphical model foundation for enabling predictive digital twins at scale," *Nature Computational Science*, Vol. 1, No. 5, 2021, pp. 337–347. DOI: 10.1038/s43588-021-00069-0 cited on pp. 4, 25, 84, 185, 189, 211

20 Sisson, W., Karve, P., and Mahadevan, S., "Digital Twin Approach for Component Health-Informed Rotorcraft Flight Parameter Optimization," *AIAA Journal*, Vol. 60, No. 3, 2022, pp. 1923–1936. DOI: 10.2514/1.J060770 cited on pp. 4, 84

21 Corral-Acero, J., Margara, F., Marciniak, M., Rodero, C., Loncaric, F., Feng, Y., Gilbert, A., Fernandes, J. F., Bukhari, H. A., Wajdan, A., *et al.*, "The 'Digital Twin' to enable the vision of precision cardiology," *European Heart Journal*, Vol. 41, No. 48, 2020, pp. 4556–4564. DOI: 10.1093/eurheartj/ehaa159 cited on p. 4

22 Chaudhuri, A., Pash, G., Hormuth, D. A., Lorenzo, G., Kapteyn, M., Wu, C., Lima, E. A., Yankeelov, T. E., Willcox, K., *et al.*, "Predictive digital twin for optimizing patient-specific radiotherapy regimens under uncertainty in high-grade gliomas," *Frontiers in Artificial Intelligence*, Vol. 6, 2023. DOI: 10.3389/frai.2023.1222612 cited on p. 4

23 Bruynseels, K., Santoni de Sio, F., and Van den Hoven, J., "Digital Twins in Health Care: Ethical Implications of an Emerging Engineering Paradigm," *Frontiers in Genetics*, Vol. 9, 2018, p. 31. DOI: 10.3389/fgene.2018.00031 cited on p. 4

24 Bauer, P., Stevens, B., and Hazeleger, W., "A digital twin of Earth for the green transition," *Nature Climate Change*, Vol. 11, No. 2, 2021, pp. 80–83. DOI: 10.1038/s41558-021-00986-y cited on p. 4

25 Moya, B., Badías, A., Alfaro, I., Chinesta, F., and Cueto, E., "Digital twins that learn and correct themselves," *International Journal for Numerical Methods in Engineering*, Vol. 123, No. 13, 2022, pp. 3034–3044. DOI: 10.1002/nme.6535 cited on p. 4

26 Biondini, F. and Frangopol, D. M., "Life-Cycle Performance of Deteriorating Structural Systems under Uncertainty: Review," *Journal of Structural Engineering*, Vol. 142, No. 9, 2016, F4016001. DOI: 10.1061/(ASCE)ST.1943-541X.0001544 cited on pp. 5, 191

27 Civil Engineers, A. S. of, "Adaptive design and risk management," *Climate-Resilient Infrastructure*. Reston, Virginia: American Society of Civil Engineers, 2018, ch. 7, pp. 173–226. DOI: 10.1061/9780784415191.ch07 cited on p. 5

28 Mariniello, G., Pastore, T., Menna, C., Festa, P., and Asprone, D., "Escalating impacts of climate extremes on critical infrastructures in Europe," *Global Environmental Change*, Vol. 48, 2018, pp. 97–107. DOI: 10.1016/j.gloenvcha.2017.11.007 cited on p. 5

29 Thoft-Christensen, P., "Life-cycle cost-benefit (LCCB) analysis of bridges from a user and social point of view," *Structure and Infrastructure Engineering*, Vol. 5, No. 1, 2009, pp. 49–57. DOI: 10.1080/15732470701322818 cited on p. 5

30 Jones, C., Lethco, T., Curley, J., and Zackin, D., "A preventable crisis: The economic and human costs of a Hudson River rail tunnel shutdown," Regional Plan Association, 2019. cited on p. 5

31 Gkoumas, K., Marques Dos Santos, F., Van Balen, M., Tsakalidis, A., Ortega Hortelano, A., Grosso, M., Haq, G., and Pekár, F., "Research and innovation in bridge maintenance, inspection and monitoring," Publications Office of the European Union, 2019. DOI: 10.2760/16174 cited on p. 5

32 ASCE, "A comprehensive assessment of America's infrastructure," World Headquarters: Reston, Virginia, 2021. cited on p. 5

33 Glaser, S. D. and Tolman, A., "Sense of Sensing: From Data to Informed Decisions for the Built Environment," *Journal of Infrastructure Systems*, Vol. 14, No. 1, 2008, pp. 4–14. DOI: 10.1061/(ASCE)1076-0342(2008)14:1(4) cited on p. 5

34 Achenbach, J. D., "Structural health monitoring – What is the prescription?" *Mechanics Research Communications*, Vol. 36, No. 2, 2009, pp. 137–142, ISSN: 0093-6413. DOI: 10.1016/j.mechrescom.2008.08.011 cited on p. 5

35 Kamariotis, A., Chatzi, E., and Straub, D., "Value of information from vibration-based structural health monitoring extracted via Bayesian model updating," *Mechanical Systems and Signal Processing*, Vol. 166, 2022, p. 108 465. DOI: 10.1016/j.ymssp.2021.108465 cited on p. 6

36 Andriotis, C. P., Papakonstantinou, K. G., and Chatzi, E. N., "Value of structural health information in partially observable stochastic environments," *Structural Safety*, Vol. 93, 2021, p. 102 072.
DOI: 10.1016/j.strusafe.2020.102072 cited on p. 6

37 Malings, C. and Pozzi, M., "Value-of-information in spatio-temporal systems: Sensor placement and scheduling," *Reliability Engineering & System Safety*, Vol. 172, 2018, pp. 45–57.
DOI: 10.1016/j.ress.2017.11.019 cited on p. 6

38 Farrar, C. and Worden, K., *Structural Health Monitoring: A Machine Learning Perspective*. John Wiley & Sons, 2013.
DOI: 10.1002/9781118443118 cited on pp. 7, 9, 14

39 Rytter, A., "Vibrational Based Inspection of Civil Engineering Structures," Ph.D. Dissertation, University of Aalborg, Denmark, 1993. cited on p. 7

40 Oden, J. T., Babuška, I., and Faghihi, D., "Predictive computational science: computer predictions in the presence of uncertainty," *Encyclopedia of Computational Mechanics Second Edition*. John Wiley & Sons, 2017, pp. 1–26.
DOI: 10.1002/9781119176817.ecm2101 cited on p. 8

41 Babuška, I. and Strouboulis, T., *The Finite Element Method and Its Reliability*, ser. Numerical mathematics and scientific computation. Clarendon Press, 2001. cited on p. 8

42 Taddei, T., Penn, J. D., Yano, M., and Patera, A. T., "Simulation-Based Classification; a Model-Order-Reduction Approach for Structural Health Monitoring," *Archives of Computational Methods in Engineering*, Vol. 25, 2018, pp. 23–45.
DOI: 10.1007/s11831-016-9185-0 cited on p. 8

43 Eftekhar Azam, S. and Mariani, S., "Online damage detection in structural systems via dynamic inverse analysis: A recursive Bayesian approach," *Engineering Structures*, Vol. 159, 2018, pp. 28–45.
DOI: 10.1016/j.engstruct.2017.12.031 cited on pp. 8, 216

44 Pandey, A. and Biswas, M., "Damage Detection in Structures Using Changes in Flexibility," *Journal of Sound and Vibration*, Vol. 169, No. 1, 1994, pp. 3–17.
DOI: 10.1006/jsvi.1994.1002 cited on p. 8

45 Mariniello, G., Pastore, T., Menna, C., Festa, P., and Asprone, D., "Structural damage detection and localization using decision tree ensemble and vibration data," *Computer-Aided Civil and Infrastructure Engineering*, Vol. 36, No. 9, 2021, pp. 1129–1149.
DOI: 10.1111/mice.12633 cited on pp. 8, 13

46 Kapteyn, M. G., Knezevic, D. J., Huynh, D., Tran, M., and Willcox, K. E., "Data-driven physics-based digital twins via a library of component-based reduced-order models," *International Journal for Numerical Methods in Engineering*, Vol. 123, No. 13, 2020, pp. 2986–3003.
DOI: 10.1002/nme.6423 cited on p. 8

47 Teughels, A., Maeck, J., and De Roeck, G., "Damage assessment by FE model updating using damage functions," *Computers & Structures*, Vol. 80, No. 25, 2002, pp. 1869–1879.
DOI: 10.1016/S0045-7949(02)00217-1 cited on p. 8

48 Giglioni, V., Poole, J., Venanzi, I., Ubertini, F., and Worden, K., "A domain adaptation approach to damage classification with an application to bridge monitoring," *Mechanical Systems and Signal Processing*, Vol. 209, 2024, p. 111 135.
DOI: 10.1016/j.ymssp.2024.111135 cited on pp. 8, 16, 220

49 Hernández-González, I. and García-Macías, E., "Towards a comprehensive damage identification of structures through populations of competing models," *Engineering with Computers*, No. 1, 2024, pp. 85–91.
DOI: 10.1007/s00366-024-01972-6 cited on p. 9

50 Seventekidis, P., Giagopoulos, D., Arailopoulos, A., and Markogiannaki, O., "Structural Health Monitoring using deep learning with optimal finite element model generated data," *Mechanical Systems and Signal Processing*, Vol. 145, 2020, p. 106 972.
DOI: 10.1016/j.ymssp.2020.106972 cited on p. 9

51 Fernandez-Navamuel, A., Zamora-Sánchez, D., Omella, Á. J., Pardo, D., Garcia-Sanchez, D., and Magalhães, F., "Supervised Deep Learning with Finite Element simulations for damage identification in bridges," *Engineering Structures*, Vol. 257, 2022, p. 114 016.
DOI: 10.1016/j.engstruct.2022.114016 cited on p. 9

52 Ebrahimian, H., Astroza, R., Conte, J. P., and de Callafon, R. A., "Nonlinear finite element model updating for damage identification of civil structures using batch Bayesian estimation," *Mechanical Systems and Signal Processing*, Vol. 84, 2017, pp. 194–222.
DOI: 10.1016/j.ymssp.2016.02.002 cited on p. 9

53 Giagopoulos, D., Arailopoulos, A., Dertimanis, V., Papadimitriou, C., Chatzi, E., and Grompanopoulos, K., "Structural health monitoring and fatigue damage estimation using vibration measurements and finite element model updating," *Structural Health Monitoring*, Vol. 18, No. 4, 2019, pp. 1189–1206.
DOI: 10.1177/1475921718790188 cited on p. 9

54 Luca Rosfalco, "Blending physics and data in structural health monitoring," Ph.D. Dissertation, Politecnico di Milano, 2022. cited on p. 9

55 Biondini, F. and Vergani, M., "Deteriorating beam finite element for nonlinear analysis of concrete structures under corrosion," *Structure and Infrastructure Engineering*, Vol. 11, No. 4, 2015, pp. 519–532.
DOI: 10.1080/15732479.2014.951863 cited on p. 9

56 Cristiani, D., Sbarufatti, C., Cadini, F., and Giglio, M., "Fatigue damage diagnosis and prognosis of an aeronautical structure based on surrogate modelling and particle filter," *Structural Health Monitoring*, Vol. 20, No. 5, 2021, pp. 2726–2746.
DOI: 10.1177/1475921720971551 cited on p. 9

57 Eftekhar Azam, S., Mariani, S., and Attari, N., "Online damage detection via a synergy of proper orthogonal decomposition and recursive Bayesian filters," *Nonlinear Dynamics*, Vol. 89, No. 2, 2017, pp. 1489–1511.
DOI: 10.1007/s11071-017-3530-1 cited on pp. 9, 109

58 Corigliano, A. and Mariani, S., "Parameter identification in explicit structural dynamics: Performance of the extended Kalman filter," *Computer Methods in Applied Mechanics and Engineering*, Vol. 193, No. 36, 2004, pp. 3807–3835.
DOI: 10.1016/j.cma.2004.02.003 cited on pp. 9, 109

59 Rosafalco, L., Eftekhar Azam, S., Mariani, S., and Corigliano, A., "System Identification via Unscented Kalman Filtering and Model Class Selection," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, Vol. 10, No. 1, 2024, p. 04 023 063.
DOI: 10.1061/AJRUA6.RUENG-1085 cited on p. 9

60 Hou, R., Wang, X., and Xia, Y., *Vibration-based structural damage detection using sparse bayesian learning techniques*, *Structural Health Monitoring Based on Data Science Techniques*, Springer International Publishing, 2022, pp. 1–25.
DOI: 10.1007/978-3-030-81716-9_1 cited on p. 9

61 Green, P. L. and Worden, K., "Bayesian and Markov chain Monte Carlo methods for identifying nonlinear systems in the presence of uncertainty," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 373, No. 2051, 2015, p. 20 140 405.
DOI: 10.1098/rsta.2014.0405 cited on p. 9

62 Ierimonti, L., Cavalagli, N., García-Macías, E., Venanzi, I., and Ubertini, F., *Bayesian-based damage assessment of historical structures using vibration monitoring data*, *International Workshop on Civil* cited on p. 9

Structural Health Monitoring, Springer, 2021, pp. 415–429.
DOI: 10.1007/978-3-030-74258-4_28

63 Lye, A., Cicirello, A., and Patelli, E., "A review of stochastic sampling methods for Bayesian inference problems," *Proceedings of the 29th European Safety and Reliability Conference*, Research Publishing Services, 2019.
DOI: 10.3850/978-981-11-2724-3-1087-cd cited on p. 9

64 Muto, M. and Beck, J. L., "Bayesian updating and model class selection for hysteretic structural models using stochastic simulation," *Journal of Vibration and Control*, Vol. 14, No. 1–2, 2008, pp. 7–34.
DOI: 10.1177/1077546307079400 cited on p. 9

65 Gomes, G. F., Cunha, S. S. da, and Ancelotti, A. C., "A sunflower optimization (SFO) algorithm applied to damage identification on laminated composite plates," *Engineering with Computers*, Vol. 35, No. 2, 2019, pp. 619–626.
DOI: 10.1007/s00366-018-0620-8 cited on p. 9

66 Tran-Ngoc, H., Khatir, S., De Roeck, G., Bui-Tien, T., Nguyen-Ngoc, L., and Abdel Wahab, M., "Model Updating for Nam O Bridge Using Particle Swarm Optimization Algorithm and Genetic Algorithm," *Sensors*, Vol. 18, No. 12, 2018, p. 4131.
DOI: 10.3390/s18124131 cited on p. 9

67 Tiachacht, S., Bouazzouni, A., Khatir, S., Abdel Wahab, M., Behtani, A., and Capozucca, R., "Damage assessment in structures using combination of a modified Cornwell indicator and genetic algorithm," *Engineering Structures*, Vol. 177, 2018, pp. 421–430.
DOI: 10.1016/j.engstruct.2018.09.070 cited on p. 9

68 Ferrari, R., Froio, D., Rizzi, E., Gentile, C., and Chatzi, E. N., "Model updating of a historic concrete bridge by sensitivity- and global optimization-based Latin Hypercube Sampling," *Engineering Structures*, Vol. 179, 2019, pp. 139–160.
DOI: 10.1016/j.engstruct.2018.08.004 cited on p. 9

69 Kabanikhin, S. I., "Definitions and examples of inverse and ill-posed problems," *Journal of Inverse and Ill-posed Problems*, Vol. 16, No. 4, 2008, pp. 317–357.
DOI: 10.1515/JIIP.2008.019 cited on p. 10

70 Bishop, C. M., *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. New York, New York: Springer, 2006. cited on pp. 10, 34, 61, 63

71 Farrar, C. R., Doebling, S. W., and Nix, D. A., "Vibration–Based Structural Damage Identification," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 359, No. 1778, 2001, pp. 131–149.
DOI: 10.1098/rsta.2000.0717 cited on pp. 10, 59

72 W. Doebling, S., Farrar, C., and Prime, M., "A Summary Review of Vibration–Based Damage Identification Methods," *The Shock and Vibration Digest*, Vol. 30, No. 2, 1998, pp. 91–105.
DOI: 10.1177/058310249803000201 cited on p. 11

73 Buckley, T., Ghosh, B., and Pakrashi, V., "A Feature Extraction & Selection Benchmark for Structural Health Monitoring," *Structural Health Monitoring*, Vol. 22, No. 3, 2023, pp. 2082–2127.
DOI: 10.1177/14759217221111141 cited on p. 11

74 Wang, Y., Khoo, S., Li, A.-.-J., and Hao, H., "FEM Calibrated ARMAX Model Updating Method for Time Domain Damage Identification," *Advances in Structural Engineering*, Vol. 16, No. 1, 2013, pp. 51–60.
DOI: 10.1260/1369-4332.16.1.51 cited on p. 11

75 Entezami, A. and Shariatmadar, H., "An unsupervised learning approach by novel damage indices in structural health monitoring for damage localization and quantification," *Structural Health Monitoring*, Vol. 17, No. 2, 2018, pp. 325–345.
DOI: 10.1177/1475921717693572 cited on p. 11

76 Giglioni, V., García-Macías, E., Venanzi, I., Ierimonti, L., and Ubertini, F., "The use of receiver operating characteristic curves and precision-versus-recall curves as performance metrics in unsupervised structural damage classification under changing environment," *Engineering Structures*, Vol. 246, 2021, p. 113 029.
DOI: 10.1016/j.engstruct.2021.113029 cited on p. 12

77 Entezami, A., Sarmadi, H., Behkamal, B., and Mariani, S., "Big data analytics and structural health monitoring: A statistical pattern recognition-based approach," *Sensors*, Vol. 20, No. 8, 2020, p. 2328.
DOI: 10.3390/s20082328 cited on p. 12

78 Entezami, A., Shariatmadar, H., and Mariani, S., "Early damage assessment in large-scale structures by innovative statistical pattern recognition methods based on time series modeling and novelty detection," *Advances in Engineering Software*, Vol. 150, 2020, p. 102 923.
DOI: 10.1016/j.advengsoft.2020.102923 cited on p. 12

79 Yan, W.-J., Chronopoulos, D., Yuen, K.-V., and Zhu, Y.-C., "Structural anomaly detection based on probabilistic distance measures of transmissibility function and statistical threshold selection scheme," *Mechanical Systems and Signal Processing*, Vol. 162, 2022, p. 108 009. cited on p. 12
DOI: 10.1016/j.ymssp.2021.108009

80 Worden, K., Manson, G., and Fieller, N. R., "Damage detection using outlier analysis," *Journal of Sound and Vibration*, Vol. 229, No. 3, 2000, pp. 647–667. cited on p. 12
DOI: 10.1006/jsvi.1999.2514

81 Bull, L., Worden, K., Fuentes, R., Manson, G., Cross, E., and Dervilis, N., "Outlier ensembles: A robust method for damage detection and unsupervised feature extraction from high-dimensional data," *Journal of Sound and Vibration*, Vol. 453, 2019, pp. 126–150. cited on p. 12
DOI: 10.1016/j.jsv.2019.03.025

82 Liu, F. T., Ting, K. M., and Zhou, Z.-H., "Isolation Forest," *8th IEEE International Conference on Data Mining*, 2008, pp. 413–422. cited on p. 12
DOI: 10.1109/ICDM.2008.17

83 Santos, A., Figueiredo, E., Silva, M., Sales, C., and Costa, J., "Machine learning algorithms for damage detection: Kernel-based approaches," *Journal of Sound and Vibration*, Vol. 363, 2016, pp. 584–599. cited on p. 12
DOI: 10.1016/j.jsv.2015.11.008

84 Goldstein, M. and Uchida, S., "A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data," *PLoS One*, Vol. 11, 2016, pp. 1–31. cited on p. 12
DOI: 10.1371/journal.pone.0152173

85 Diez, A., Khoa, N. L. D., Makki Alamdari, M., Wang, Y., Chen, F., and Runcie, P., "A clustering approach for structural health monitoring on bridges," *Journal of Civil Structural Health Monitoring*, Vol. 6, 2016, pp. 429–445. cited on p. 12
DOI: 10.1007/s13349-016-0160-0

86 Wang, Z. and Cha, Y.-j., "Automated damage-sensitive feature extraction using unsupervised convolutional neural networks," *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018*, Sohn, H., Ed. SPIE, 2018, Vol. 10598, 105981J. cited on p. 12
DOI: 10.1117/12.2295966

87 Bigoni, C. and Hesthaven, J. S., "Simulation-based Anomaly Detection and Damage Localization: An application to Structural Health Monitoring," *Computer Methods in Applied Mechanics and* cited on p. 12

Engineering, Vol. 363, 2020, p. 112 896.
DOI: 10.1016/j.cma.2020.112896

88 Cha, Y.-J. and Wang, Z., "Unsupervised novelty detection-based structural damage localization using a density peaks-based fast clustering algorithm," *Structural Health Monitoring*, Vol. 17, No. 2, 2018, pp. 313–324.
DOI: 10.1177/1475921717691260 cited on p. 12

89 Entezami, A., Shariatmadar, H., and Mariani, S., "Fast unsupervised learning methods for structural health monitoring with large vibration data from dense sensor networks," *Structural Health Monitoring*, Vol. 19, No. 6, 2020, pp. 1685–1710.
DOI: 10.1177/1475921719894186 cited on p. 12

90 Worden, K., Farrar, C. R., Manson, G., and Park, G., "The Fundamental Axioms of Structural Health Monitoring," *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 463, No. 2082, 2007, pp. 1639–1664.
DOI: 10.1098/rspa.2007.1834 cited on pp. 12, 14

91 Yuan, F.-G., Zargar, S. A., Chen, Q., and Wang, S., "Machine learning for structural health monitoring: challenges and opportunities," *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2020*, Huang, H., Ed., Vol. 11379, International Society for Optics and Photonics, SPIE, 2020, p. 1 137 903.
DOI: 10.1117/12.2561610 cited on p. 12

92 Ritto, T. and Rochinha, F., "Digital twin, physics-based model, and machine learning applied to damage detection in structures," *Mechanical Systems and Signal Processing*, Vol. 155, 2021, p. 107 614.
DOI: 10.1016/j.ymssp.2021.107614 cited on p. 13

93 Kapteyn, M. G., Knezevic, D. J., and Willcox, K., "Toward predictive digital twins via component-based reduced-order models and interpretable machine learning," *AIAA Scitech 2020 Forum*, p. 0418.
DOI: 10.2514/6.2020-0418 cited on p. 13

94 Rosafalco, L., Torzoni, M., Manzoni, A., Mariani, S., and Corigliano, A., "Online structural health monitoring by model order reduction and deep learning algorithms," *Computers & Structures*, Vol. 255, 2021, p. 106 604.
DOI: 10.1016/j.compstruc.2021.106604 cited on pp. 13, 45, 131

95 Abdeljaber, O., Avci, O., Kiranyaz, S., Gabbouj, M., and Inman, D. J., "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks," *Journal of Sound and Vibration*, Vol. 388, 2017, pp. 154–170.
DOI: 10.1016/j.jsv.2016.10.043 cited on pp. 13, 77

96 Ghiasi, R., Torkzadeh, P., and Noori, M., "A machine-learning approach for structural damage detection using least square support vector machine based on a new combinational kernel function," *Structural Health Monitoring*, Vol. 15, No. 3, 2016, pp. 302–316.
DOI: 10.1177/1475921716639587 cited on p. 13

97 Sajedi, S. O. and Liang, X., "Vibration-based semantic damage segmentation for large-scale structural health monitoring," *Computer-Aided Civil and Infrastructure Engineering*, Vol. 35, No. 6, 2020, pp. 579–596.
DOI: 10.1111/mice.12523 cited on p. 13

98 Gulgec, N., Takáč, M., and Pakzad, S., "Convolutional Neural Network Approach for Robust Structural Damage Detection and Localization," *Journal of Computing in Civil Engineering*, Vol. 33, No. 3, 2019, p. 04 019 005.
DOI: 10.1061/(ASCE)CP.1943-5487.0000820 cited on p. 13

99 Fernandez-Navamuel, A., Pardo, D., Magalhães, F., Zamora-Sánchez, D., Omella, Á. J., and Garcia-Sanchez, D., "Bridge damage identification under varying environmental and operational conditions combining Deep Learning and numerical simulations," *Mechanical Systems and Signal Processing*, Vol. 200, 2023, p. 110 471.
DOI: 10.1016/j.ymssp.2023.110471 cited on p. 13

100 Rosafalco, L., Torzoni, M., Manzoni, A., Mariani, S., and Corigliano, A., "A self-adaptive hybrid model/data-driven approach to shm based on model order reduction and deep learning," *Structural Health Monitoring Based on Data Science Techniques*, Cury, A., Ribeiro, D., Ubertini, F., and Todd, M. D., Eds. Cham, Switzerland: Springer International Publishing, 2022, pp. 165–184.
DOI: 10.1007/978-3-030-81716_9 cited on pp. 13, 45

101 Li, X., Xi, H., Zhou, C., Gu, W., and Gao, T., "Damage Degree Identification of Crane Girder Based on the Support Vector Machine," *Prognostics and System Health Management Conference*, 2018, pp. 920–924.
DOI: 10.1109/PHM-Chongqing.2018.00164 cited on p. 13

102 LeCun, Y., Bengio, Y., and Hinton, G., "Deep learning," *Nature*, Vol. 521, 2015, pp. 436–444.
DOI: 10.1038/nature14539 cited on pp. 13, 65

103 Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*. Cambridge, Massachusetts: MIT Press, 2016. cited on pp. 13, 61, 63, 66, 69, 71, 77, 78, 79

104 Fink, O., Wang, Q., Svensen, M., Dersin, P., Lee, W.-J., and Ducoffe, M., "Potential, Challenges and Future Directions for Deep Learning in Prognostics and Health Management Applications," *Engineering Applications of Artificial Intelligence*, Vol. 92, 2020, p. 103 678.
DOI: 10.1016/j.engappai.2020.103678 cited on pp. 13, 66

105 Torzoni, M., Rosafalco, L., and Manzoni, A., "A Combined Model-Order Reduction and Deep Learning Approach for Structural Health Monitoring under Varying Operational and Environmental Conditions," *Engineering Proceedings*, Vol. 2, No. 1, 2020, p. 94.
DOI: 10.3390/ecsa-7-08258 cited on pp. 13, 16, 45

106 Torzoni, M., Rosafalco, L., Manzoni, A., Mariani, S., and Corigliano, A., "SHM under varying environmental conditions: An approach based on model order reduction and deep learning," *Computers & Structures*, Vol. 266, 2022, p. 106 790.
DOI: 10.1016/j.compstruc.2022.106790 cited on pp. 13, 16, 45, 217

107 Alampalli, S., "Influence of in–service environment on modal parameters," *Proceedings–SPIE the International Society for Optical Engineering*, Vol. 1, 1998, pp. 111–116. cited on p. 13

108 Sohn, H., Worden, K., and Farrar, C. R., "Statistical Damage Classification Under Changing Environmental and Operational Conditions," *Journal of Intelligent Material Systems and Structures*, Vol. 13, No. 9, 2002, pp. 561–574.
DOI: 10.1106/104538902030904 cited on pp. 13, 15, 81

109 Farrar, C. R., Doebling, S. W., Cornwell, P. J., and Straser, E. G., "Variability of modal parameters measured on the Alamosa Canyon Bridge," Los Alamos National Lab., Los Alamos, New Mexico, 1996. cited on p. 15

110 Worden, K. and Cross, E., "On switching response surface models, with applications to the structural health monitoring of bridges," *Mechanical Systems and Signal Processing*, Vol. 98, 2018, pp. 139–156.
DOI: 10.1016/j.ymssp.2017.04.022 cited on p. 15

111 Cross, E., Koo, K., Brownjohn, J., and Worden, K., "Long-term monitoring and data analysis of the Tamar Bridge," *Mechanical Systems and Signal Processing*, Vol. 35, No. 1, 2013, pp. 16–34.
DOI: 10.1016/j.ymssp.2012.08.026 cited on p. 15

112 Sohn, H., Dzwonczyk, M., Straser, E. G., Kiremidjian, A. S., Law, K. H., and Meng, T., "An experimental study of temperature effect on modal parameters of the Alamosa Canyon Bridge," *Earthquake Engineering & Structural Dynamics*, Vol. 28, No. 8, 1999, pp. 879–897. cited on p. 15

DOI: 10.1002/(SICI)1096-9845(199908)28:8<879::AID-EQE845>3.0.CO;2-V

113 Manson, G., "Identifying damage sensitive, environment insensitive features for damage detection," *Proceeding of the 3rd international conference on Identification in engineering systems*, University of Wales, Swansea, 2002, pp. 187–197. cited on p. 15

114 Cross, E., Manson, G., Worden, K., and Pierce, S., "Features for damage detection with insensitivity to environmental and operational variations," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 468, No. 2148, December 2012, pp. 4098–4122.
DOI: 10.1098/rspa.2012.0031 cited on p. 15

115 Coletta, G., Miraglia, G., Pecorelli, M., Ceravolo, R., Cross, E., Surace, C., and Worden, K., "Use of the cointegration strategies to remove environmental effects from data acquired on historical buildings," *Engineering Structures*, Vol. 183, 2019, pp. 1014–1026.
DOI: 10.1016/j.engstruct.2018.12.044 cited on p. 15

116 Bull, L., Gardner, P., Gosliga, J., Rogers, T., Dervilis, N., Cross, E., Papatheou, E., Maguire, A., Campos, C., and Worden, K., "Foundations of population-based SHM, Part I: Homogeneous populations and forms," *Mechanical Systems and Signal Processing*, Vol. 148, 2021, p. 107 141.
DOI: 10.1016/j.ymssp.2020.107141 cited on pp. 16, 220

117 Gosliga, J., Gardner, P., Bull, L., Dervilis, N., and Worden, K., "Foundations of Population-based SHM, Part II: Heterogeneous populations – Graphs, networks, and communities," *Mechanical Systems and Signal Processing*, Vol. 148, 2021, p. 107 144.
DOI: 10.1016/j.ymssp.2020.107144 cited on p. 16

118 Gardner, P., Bull, L., Gosliga, J., Dervilis, N., and Worden, K., "Foundations of population-based SHM, Part III: Heterogeneous populations – Mapping and transfer," *Mechanical Systems and Signal Processing*, Vol. 149, 2021, p. 107 142.
DOI: 10.1016/j.ymssp.2020.107142 cited on pp. 16, 220

119 Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T., "Adversarial Discriminative Domain Adaptation," *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2962–2971.
DOI: 10.1109/CVPR.2017.316 cited on p. 16

120 Wang, X. and Xia, Y., "Knowledge transfer for structural damage detection through re-weighted adversarial domain adaptation," *Mechanical Systems and Signal Processing*, Vol. 172, 2022, p. 108 991.
DOI: 10.1016/j.ymssp.2022.108991 cited on pp. 16, 220

121 Gardner, P., Liu, X., and Worden, K., "On the application of domain adaptation in structural health monitoring," *Mechanical Systems and Signal Processing*, Vol. 138, 2020, p. 106 550.
DOI: 10.1016/j.ymssp.2019.106550 cited on p. 16

122 Poole, J., Gardner, P., Dervilis, N., Bull, L., and Worden, K., "On statistic alignment for domain adaptation in structural health monitoring," *Structural Health Monitoring*, Vol. 22, No. 3, 2023, pp. 1581–1600.
DOI: 10.1177/14759217221110441 cited on pp. 16, 220

123 Peherstorfer, B., Willcox, K., and Gunzburger, M., "Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization," *SIAM Review*, Vol. 60, No. 3, 2018, pp. 550–591.
DOI: 10.1137/16M1082469 cited on p. 17

124 Kerschen, G., Golinval, J.-c., Vakakis, A. F., and Bergman, L. A., "The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview," *Nonlinear Dynamics*, Vol. 41, No. 1, 2005, pp. 147–169.
DOI: 10.1007/s11071-005-2803-2 cited on pp. 18, 49, 50

125 Quarteroni, A., Manzoni, A., and Negri, F., *Reduced Basis Methods for Partial Differential Equations: An Introduction*. Springer, 2015, Vol. 92.
DOI: 10.1007/978-3-319-15431-2 cited on pp. 18, 49

126 Ladevèze, P., *Nonlinear Computational Structural Mechanics: New Approaches and Non-Incremental Methods of Calculation*. New York, New York: Springer, 1999.
DOI: 10.1007/978-1-4612-1432-8 cited on p. 18

127 Chinesta, F., Ladevèze, P., and Cueto, E., "A Short Review in Model Order Reduction Based on Proper Generalized Decomposition," *Archives of Computational Methods in Engineering*, Vol. 18, No. 4, 2011, pp. 395–404.
DOI: 10.1007/s11831-011-9064-7 cited on p. 18

128 Antoulas, A. C., *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.
DOI: 10.1137/1.9780898718713 cited on p. 18

129 Lanczos, C., "An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators," *Journal of Research of the National Bureau of Standards*, Vol. Journal of Research of the National Bureau of Standards, 1950, pp. 255–282.
DOI: 10.6028/jres.045.026 cited on p. 18

130 Arnoldi, W. E., "The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem," *Quarterly of Applied Mathematics*, Vol. 9, No. 1, 1951, pp. 17–29.
DOI: 10.1090/QAM/42792 cited on p. 18

131 Sieber, M., Paschereit, C. O., and Oberleithner, K., "Spectral proper orthogonal decomposition," *Journal of Fluid Mechanics*, Vol. 792, 2016, pp. 798–828.
DOI: 10.1017/jfm.2016.103 cited on p. 18

132 Towne, A., Schmidt, O. T., and Colonius, T., "Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis," *Journal of Fluid Mechanics*, Vol. 847, 2018, pp. 821–867.
DOI: 10.1017/jfm.2018.283 cited on p. 18

133 Santos, K. R. dos, Giovanis, D. G., Kontolati, K., Loukrezis, D., and Shields, M. D., "Grassmannian diffusion maps based surrogate modeling via geometric harmonics," *International Journal for Numerical Methods in Engineering*, Vol. 123, No. 15, 2022, pp. 3507–3529.
DOI: 10.1002/nme.6977 cited on p. 18

134 Fresca, S., Dede', L., and Manzoni, A., "Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Nonlinear Time-Dependent Parametrized PDEs," *Journal of Scientific Computing*, Vol. 87, 2021, pp. 1–36.
DOI: 10.1007/s10915-021-01462-7 cited on pp. 18, 172

135 Peherstorfer, B. and Willcox, K., "Data-driven operator inference for nonintrusive projection-based model reduction," *Computer Methods in Applied Mechanics and Engineering*, Vol. 306, 2016, pp. 196–215.
DOI: 10.1016/j.cma.2016.03.025 cited on p. 18

136 Qian, E., Farcaş, I.-G., and Willcox, K., "Reduced Operator Inference for Nonlinear Partial Differential Equations," *SIAM Journal on Scientific Computing*, Vol. 44, No. 4, 2022, A1934–A1959.
DOI: 10.1137/21M1393972 cited on p. 18

137 Geelen, R., Wright, S., and Willcox, K., "Operator inference for non-intrusive model reduction with quadratic manifolds," *Computer Methods in Applied Mechanics and Engineering*, Vol. 403, 2023, p. 115 717.
DOI: 10.1016/j.cma.2022.115717 cited on p. 18

138 Forrester, A., Sobester, A., and Keane, A., *Engineering Design Via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, 2008.
DOI: 10.1002/9780470770801 cited on p. 18

139 Koziel, S. and Leifsson, L., *Surrogate-Based Modeling and Optimization: Applications in Engineering*. New York, New York: Springer, 2013.
DOI: 10.1007/978-1-4614-7551-4 cited on p. 18

140 Maulik, R., Fukami, K., Ramachandra, N., Fukagata, K., and Taira, K., "Probabilistic neural networks for fluid flow surrogate modeling and data recovery," *Physical Review Fluids*, Vol. 5, 10 2020, p. 104 401.
DOI: 10.1103/PhysRevFluids.5.104401 cited on p. 18

141 Sadeghi Eshkevari, S., Takáč, M., Pakzad, S. N., and Jahani, M., "DynNet: Physics-based neural architecture design for nonlinear structural response modeling and prediction," *Engineering Structures*, Vol. 229, 2021, p. 111 582.
DOI: 10.1016/j.engstruct.2020.111582 cited on p. 18

142 García-Macías, E., Ierimonti, L., Venanzi, I., and Ubertini, F., "An Innovative Methodology for Online Surrogate-Based Model Updating of Historic Buildings Using Monitoring Data," *International Journal of Architectural Heritage*, Vol. 15, No. 1, 2021, pp. 92–112.
DOI: 10.1080/15583058.2019.1668495 cited on p. 18

143 Raissi, M., Perdikaris, P., and Karniadakis, G. E., "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, Vol. 378, 2019, pp. 686–707.
DOI: 10.1016/j.jcp.2018.10.045 cited on p. 18

144 Gunn, S. R., "Support Vector Machines for Classification and Regression," Project Report, Southampton, United Kingdom, 1998. cited on pp. 18, 61, 63

145 Geneva, N. and Zabaras, N., "Multi-fidelity generative deep learning turbulent flows," *Foundations of Data Science*, Vol. 2, No. 4, 2020, pp. 391–428.
DOI: 10.3934/fods.2020019 cited on p. 18

146 Zhang, X., Xie, F., Ji, T., Zhu, Z., and Zheng, Y., "Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization," *Computer Methods in Applied Mechanics and Engineering*, Vol. 373, 2021, p. 113 485.
DOI: 10.1016/j.cma.2020.113485 cited on p. 18

147 Meng, X. and Karniadakis, G. E., "A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems," *Journal of Computational Physics*, Vol. 401, 2020, p. 109 020.
DOI: 10.1016/j.jcp.2019.109020 cited on pp. 18, 137

148 Meng, X., Babaee, H., and Karniadakis, G. E., "Multi-fidelity Bayesian Neural Networks: Algorithms and Applications," *Journal of Computational Physics*, Vol. 438, 2021, p. 110 361. DOI: 10.1016/j.jcp.2021.110361 cited on p. 18

149 Guo, M., Manzoni, A., Amendt, M., Conti, P., and Hesthaven, J. S., "Multi-fidelity regression using artificial neural networks: Efficient approximation of parameter-dependent output quantities," *Computer Methods in Applied Mechanics and Engineering*, Vol. 389, 2022, p. 114 378. DOI: 10.1016/j.cma.2021.114378 cited on pp. 18, 137

150 Conti, P., Guo, M., Manzoni, A., and Hesthaven, J. S., "Multi-fidelity surrogate modeling using long short-term memory networks," *Computer Methods in Applied Mechanics and Engineering*, Vol. 404, 2023, p. 115 811. DOI: 10.1016/j.cma.2022.115811 cited on pp. 18, 78

151 Kaya, M. and Bilge, H. Ş., "Deep Metric Learning: A Survey," *Symmetry*, Vol. 11, 2019, p. 1066. DOI: 10.3390/sym11091066 cited on pp. 20, 101

152 Bellet, A., Habrard, A., and Sebban, M., "A Survey on Metric Learning for Feature Vectors and Structured Data," arXiv preprint arXiv:1306.6709, 2013 cited on pp. 20, 101

153 Cakir, F., He, K., Xia, X., Kulis, B., and Sclaroff, S., "Deep Metric Learning to Rank," *2019 IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1861–1870. DOI: 10.1109/CVPR.2019.00196 cited on pp. 20, 101

154 Hadsell, R., Chopra, S., and Lecun, Y., "Dimensionality Reduction by Learning an Invariant Mapping," *2006 IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 1735–1742. DOI: 10.1109/CVPR.2006.100 cited on pp. 20, 23, 105, 169

155 Chopra, S., Hadsell, R., and LeCun, Y., "Learning a similarity metric discriminatively, with application to face verification," *2005 IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 539–546. DOI: 10.1109/CVPR.2005.202 cited on pp. 20, 23, 105

156 Bromley, J., Guyon, I., Lecun, Y., Säckinger, E., and Shah, R., "Signature verification using a "siamese" time delay neural network," *Advances in Pattern Recognition Systems Using Neural Network Technologies*. 1993, pp. 25–44. DOI: 10.1142/9789812797926_0003 cited on pp. 20, 101, 104, 169

157 Kennedy, M. C. and O'Hagan, A., "Predicting the Output from a Complex Computer Code When Fast Approximations Are Available," *Biometrika*, Vol. 87, No. 1, 2000, pp. 1–13. DOI: 10.1093/biomet/87.1.1 cited on p. 22

158 Kast, M., Guo, M., and Hesthaven, J. S., "A non-intrusive multifidelity method for the reduced order modeling of nonlinear problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 364, 2020, p. 112 947.
DOI: 10.1016/j.cma.2020.112947 cited on p. 22

159 Goh, J., Bingham, D., Holloway, J. P., Grosskopf, M. J., Kuranz, C. C., and Rutter, E., "Prediction and Computer Model Calibration Using Outputs From Multifidelity Simulators," *Technometrics*, Vol. 55, No. 4, 2013, pp. 501–512.
DOI: 10.1080/00401706.2013.838910 cited on p. 22

160 Wang, Z. and Oates, T., "Imaging Time-Series to Improve Classification and Imputation," *Proceedings of the 24th International Conference on Artificial Intelligence*, Vol. 24, Buenos Aires, Argentina: AAAI Press, 2015, pp. 3939–3945.
DOI: 10.5555/2832747.2832798 cited on pp. 23, 165, 168, 269

161 Koller, D. and Friedman, N., *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, Massachusetts: MIT Press, 2009. cited on pp. 25, 83, 84

162 Russell, S. J., *Artificial Intelligence: A Modern Approach*. London, United Kingdom: Pearson Education, 2020. cited on pp. 25, 84, 85

163 Darema, F., "Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements," *International Conference on Computational Science*, Bubak, M., Albada, G. D. van, Sloot, P. M. A., and Dongarra, J., Eds., Berlin, Germany: Springer, 2004, pp. 662–669.
DOI: 10.1007/978-3-540-24688-6_86 cited on p. 25

164 Torzoni, M., Manzoni, A., and Mariani, S., "Structural health monitoring of civil structures: A diagnostic framework powered by deep metric learning," *Computers & Structures*, Vol. 271, 2022, p. 106 858.
DOI: 10.1016/j.compstruc.2022.106858 cited on pp. 31, 59, 101

165 Torzoni, M., Manzoni, A., and Mariani, S., "A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks," *Mechanical Systems and Signal Processing*, Vol. 197, 2023, p. 110 376.
DOI: 10.1016/j.ymssp.2023.110376 cited on pp. 31, 59, 135

166 Torzoni, M., Manzoni, A., and Mariani, S., "Enhancing Bayesian model updating in structural health monitoring via learnable mapping," under review. cited on pp. 31, 59, 165

167 Torzoni, M., Rosafalco, L., Mariani, S., Corigliano, A., and Manzoni, A., "Damage identification using physics-based datasets: from convolutional to metric-informed damage-sensitive feature extractors," *Mechanical, Aerospace and Complex Infrastructure Systems*, Noori, M., Yuan, F.-G., and Farsangi, E. N., Eds. Berlin, Boston: De Gruyter, 2023, pp. 101–124. DOI: 10.1515/9783110791426-005 cited on pp. 31, 59, 101

168 Torzoni, M., Manzoni, A., and Mariani, S., *A deep neural network, multi-fidelity surrogate model approach for bayesian model updating in shm*, *European Workshop on Structural Health Monitoring*, Rizzo, P. and Milazzo, A., Eds., Cham, Switzerland: Springer International Publishing, 2023, pp. 1076–1086. DOI: 10.1007/978-3-031-07258-1_108 cited on pp. 31, 135

169 Torzoni, M., Manzoni, A., and Mariani, S., "Health Monitoring of Civil Structures: A MCMC Approach Based on a Multi-Fidelity Deep Neural Network Surrogate," *Computer Sciences & Mathematics Forum*, Vol. 2, No. 1, 2022. DOI: 10.3390/IOCA2021-10889 cited on pp. 31, 135

170 Beck, J. L. and Katafygiotis, L. S., "Updating Models and Their Uncertainties. I: Bayesian Statistical Framework," *Journal of Engineering Mechanics*, Vol. 124, No. 4, 1998, pp. 455–461. DOI: 10.1061/(ASCE)0733-9399(1998)124:4(455) cited on p. 34

171 Katafygiotis, L. S., Papadimitriou, C., and Lam, H.-F., "A probabilistic approach to structural model updating," *Soil Dynamics and Earthquake Engineering*, Vol. 17, No. 7-8, 1998, pp. 495–507. DOI: 10.1016/S0267-7261(98)00008-6 cited on p. 34

172 Wang, X., Hou, R., Xia, Y., and Zhou, X., "Laplace approximation in sparse Bayesian learning for structural damage detection," *Mechanical Systems and Signal Processing*, Vol. 140, 2020, p. 106 701. DOI: 10.1016/j.ymssp.2020.106701 cited on p. 34

173 Li, B. and Der Kiureghian, A., "Operational modal identification using variational Bayes," *Mechanical Systems and Signal Processing*, Vol. 88, 2017, pp. 377–398. DOI: 10.1016/j.ymssp.2016.11.007 cited on p. 34

174 Baldacchino, T., Cross, E. J., Worden, K., and Rowson, J., "Variational Bayesian mixture of experts models and sensitivity analysis for nonlinear dynamical systems," *Mechanical Systems and Signal Processing*, Vol. 66, 2016, pp. 178–200. DOI: 10.1016/j.ymssp.2015.05.009 cited on p. 34

175 Lam, H. F., Yang, J. H., and Au, S. K., "Markov chain Monte Carlo-based Bayesian method for structural model updating and damage detection," *Structural Control and Health Monitoring*, cited on p. 34

Vol. 25, No. 4, 2018.
DOI: 10.1002/stc.2140

176 Lye, A., Cicirello, A., and Patelli, E., "An efficient and robust sampler for Bayesian inference: Transitional Ensemble Markov Chain Monte Carlo," *Mechanical Systems and Signal Processing*, Vol. 167, 2022, p. 108 471.
DOI: 10.1016/j.ymssp.2021.108471 cited on p. 34

177 Betz, W., Papaioannou, I., and Straub, D., "Transitional Markov Chain Monte Carlo: Observations and Improvements," *Journal of Engineering Mechanics*, Vol. 142, No. 5, 2016, p. 04 016 016.
DOI: 10.1061/(ASCE)EM.1943-7889.0001066 cited on pp. 34, 163

178 Mirzazadeh, R., Eftekhar Azam, S., and Mariani, S., "Mechanical Characterization of Polysilicon MEMS: A Hybrid TMCMC/POD-Kriging Approach," *Sensors*, Vol. 18, No. 4, 2018, p. 1243.
DOI: 10.3390/s18041243 cited on p. 34

179 Rocchetta, R., Broggi, M., Huchet, Q., and Patelli, E., "On-line Bayesian model updating for structural health monitoring," *Mechanical Systems and Signal Processing*, Vol. 103, 2018, pp. 174–195.
DOI: 10.1016/j.ymssp.2017.10.015 cited on p. 34

180 Meeds, E. and Welling, M., "GPS-ABC: Gaussian Process Surrogate Approximate Bayesian Computation," *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, Quebec City, Quebec, Canada, 2014, pp. 593–602.
DOI: 10.5555/3020751.3020813 cited on p. 34

181 Giannakeas, I. N., Sharif Khodaei, Z., and Aliabadi, M., "Digital clone testing platform for the assessment of SHM systems under uncertainty," *Mechanical Systems and Signal Processing*, Vol. 163, 2022, p. 108 150.
DOI: 10.1016/j.ymssp.2021.108150 cited on p. 34

182 Cristiani, D., Sbarufatti, C., and Giglio, M., "Damage diagnosis and prognosis in composite double cantilever beam coupons by particle filtering and surrogate modelling," *Structural Health Monitoring*, Vol. 20, No. 3, 2021, pp. 1030–1050.
DOI: 10.1177/1475921720960067 cited on p. 34

183 Eftekhar Azam, S., Chatzi, E., and Papadimitriou, C., "A dual Kalman filter approach for state estimation via output-only acceleration measurements," *Mechanical Systems and Signal Processing*, Vol. 60–61, 2015, pp. 866–886.
DOI: 10.1016/j.ymssp.2015.02.001 cited on p. 34

184 Katafygiotis, L. S. and Beck, J. L., "Updating Models and Their Uncertainties. II: Model Identifiability," *Journal of Engineering Mechanics*, Vol. 124, No. 4, 1998, pp. 463–467.
DOI: 10.1061/(ASCE)0733-9399(1998)124:4(463) cited on p. 34

185 Parisi, G., *Statistical Field Theory*. Avalon Publishing, 1998. cited on p. 34

186 Wang, Xiaoyou, "Probabilistic machine learning and Bayesian inference for vibration-based structural damage identification," Ph.D. Dissertation, Hong Kong Polytechnic University, 2022. cited on p. 34

187 Brooks, S., Gelman, A., Jones, G., and Meng, X.-L., *Handbook of Markov Chain Monte Carlo*. Chapman Hall/CRC, 2011.
DOI: 10.1201/b10905 cited on p. 34

188 Haario, H., Saksman, E., and Tamminen, J., "An Adaptive Metropolis Algorithm," *Bernoulli*, Vol. 7, No. 2, 2001, pp. 223–242.
DOI: 10.2307/3318737 cited on pp. 34, 40, 158

189 Geman, S. and Geman, D., "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 6, 1984, pp. 721–741.
DOI: 10.1109/TPAMI.1984.4767596 cited on pp. 34, 173

190 Ching, J. and Chen, Y.-C., "Transitional Markov Chain Monte Carlo Method for Bayesian Model Updating, Model Class Selection, and Model Averaging," *Journal of Engineering Mechanics*, Vol. 133, No. 7, 2007, pp. 816–832.
DOI: 10.1061/(ASCE)0733-9399(2007)133:7(816) cited on pp. 34, 173

191 Green, P. J. and Mira, A., "Delayed Rejection in Reversible Jump Metropolis-Hastings," *Biometrika*, Vol. 88, No. 4, 2001, pp. 1035–1053.
DOI: 10.1093/biomet/88.4.1035 cited on pp. 34, 173

192 Hoffman, M. D. and Gelman, A., "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo," *Journal of Machine Learning Research*, Vol. 15, No. 1, 2014, pp. 1593–1623.
DOI: 10.5555/2627435.2638586 cited on pp. 34, 163, 173

193 Mariani, S. and Corigliano, A., "Impact induced composite delamination: state and parameter identification via joint and dual extended Kalman filters," *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, No. 50, 2005, pp. 5242–5272.
DOI: 10.1016/j.cma.2005.01.007 cited on p. 35

194 Bolzon, G., Fedele, R., and Maier, G., "Parameter identification of a cohesive crack model by Kalman filter," *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, No. 25, 2002, pp. 2847–2871.
DOI: 10.1016/S0045-7825(02)00223-2 cited on p. 35

195 Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, Vol. 82, No. 1, 1960, pp. 35–45.
DOI: 10.1115/1.3662552 cited on p. 35

196 Goulet, J.-A., *Probabilistic Machine Learning for Civil Engineers*. MIT Press, 2020. cited on pp. 35, 41, 61, 63

197 Kaminski, P., Bryson, A., and Schmidt, S., "Discrete square root filtering: A survey of current techniques," *IEEE Transactions on automatic control*, Vol. 16, No. 6, 1971, pp. 727–736.
DOI: 10.1109/TAC.1971.1099816 cited on p. 35

198 Murphy, K. P., *Machine learning: A probabilistic perspective*. MIT press, 2012. cited on p. 35

199 Asch, M., Bocquet, M., and Nodet, M., *Data assimilation: methods, algorithms, and applications*, ser. Fundamentals of Algorithms. SIAM, 2016. cited on p. 35

200 Särkkä, S. and Svensson, L., *Bayesian Filtering and Smoothing*. Cambridge, United Kingdom: Cambridge University Press, 2013, Vol. 17.
DOI: 10.1017/CBO9781139344203 cited on p. 35

201 Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E., "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, Vol. 21, No. 6, 1953, pp. 1087–1092.
DOI: 10.1063/1.1699114 cited on pp. 37, 156

202 Hastings, W. K., "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, Vol. 57, No. 1, 1970, pp. 97–109.
DOI: 10.2307/2334940 cited on p. 37

203 Gelman, A. and Rubin, D. B., "Inference from Iterative Simulation Using Multiple Sequences," *Statistical Science*, Vol. 7, No. 4, 1992, pp. 457–472.
DOI: 10.1214/ss/1177011136 cited on pp. 38, 39

204 Gelman, A. and Rubin, D. B., "A single series from the Gibbs sampler provides a false sense of security," *Bayesian Statistics*, Vol. 4, No. 1, 1992, pp. 625–631. cited on p. 38

205 Andrieu, C. and Thoms, J., "A tutorial on adaptive MCMC," *Statistics and Computing*, Vol. 18, No. 4, 2008, pp. 343–373.
DOI: 10.1007/s11222-008-9110-y cited on p. 40

206 Haario, H., Saksman, E., and Tamminen, J., "Adaptive proposal distribution for random walk Metropolis algorithm," *Computational Statistics*, Vol. 14, 3 1999, pp. 375–395.
DOI: 10.1007/s001800050022 cited on p. 40

207 Fix, E. and Hodges, J. L., "Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties," *International Statistical Review*, Vol. 57, No. 3, 1989, pp. 238–247. cited on p. 40
DOI: 10.2307/1403797

208 Altman, N. S., "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression," *The American Statistician*, Vol. 46, No. 3, 1992, pp. 175–185. cited on p. 40
DOI: 10.2307/2685209

209 Cover, T. and Hart, P., "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, Vol. 13, No. 1, 1967, pp. 21–27. cited on p. 40
DOI: 10.1109/TIT.1967.1053964

210 Parola, M., Galatolo, F. A., Torzoni, M., and Cimino, M. G. C. A., "Convolutional Neural Networks for Structural Damage Localization on Digital Twins," *Deep Learning Theory and Applications*, Fred, A., Sansone, C., Gusikhin, O., and Madani, K., Eds., Cham, Switzerland: Springer International Publishing, 2023, pp. 78–97. cited on p. 45
DOI: 10.1007/978-3-031-37317-6_5

211 Hughes, T. J., *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, ser. Dover Civil and Mechanical Engineering. Dover Publications, 2000. cited on p. 47

212 Newmark, N. M., "A Method of Computation for Structural Dynamics," *Journal of Engineering Mechanics*, Vol. 85, No. 3, 1959, pp. 67–94. cited on p. 47
DOI: 10.1061/JMCEA3.0000098

213 Barrault, M., Maday, Y., Nguyen, N. C., and Patera, A. T., "An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations," *Comptes Rendus Mathematique*, Vol. 339, No. 9, 2004, pp. 667–672. cited on p. 48
DOI: https://doi.org/10.1016/j.crma.2004.08.006

214 Hesthaven, J. S. and Ubbiali, S., "Non-intrusive reduced order modeling of nonlinear problems using neural networks," *Journal of Computational Physics*, Vol. 363, 2018, pp. 55–78. cited on p. 48
DOI: 10.1016/j.jcp.2018.02.037

215 Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S., *Global sensitivity analysis: The primer*. John Wiley & Sons, 2008. cited on p. 48
DOI: 10.1002/9780470725184

216 McKay, M. D., Beckman, R. J., and Conover, W. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, pp. 239–245. cited on p. 48
DOI: 10.2307/12685

217 Benner, P., Gugercin, S., and Willcox, K., "A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems," *SIAM Review*, Vol. 57, No. 4, 2015, pp. 483–531. DOI: 10.1137/130932715 cited on p. 49

218 Quarteroni, A., Rozza, G., *et al.*, *Reduced Order Methods for Modeling and Computational Reduction*. Cham, Switzerland: Springer, 2014, Vol. 9. DOI: 10.1007/978-3-319-02090-7 cited on p. 49

219 Chinesta, F., Huerta, A., Rozza, G., and Willcox, K., "Model reduction methods," *Encyclopedia of Computational Mechanics Second Edition*. John Wiley & Sons, 2017, pp. 1–36, ISBN: 9781119176817 . DOI: 10.1002/9781119176817.ecm2110 cited on p. 49

220 Sirovich, L., "Turbulence and the dynamics of coherent structures. I. Coherent structures," *Quarterly of Applied Mathematics*, Vol. 45, No. 3, 1987, pp. 561–571. DOI: 10.1090/qam/910462 cited on p. 49

221 Golub, G. H. and Van Loan, C. F., *Matrix computations*. Baltimore, Maryland: JHU press, 2013. cited on p. 49

222 Hotelling, H., "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, Vol. 24, No. 6, 1933, pp. 417–441. DOI: 10.1037/h0071325 cited on p. 50

223 Pearson, K., "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Vol. 2, No. 11, 1901, pp. 559–572. DOI: 10.1080/14786440109462720 cited on p. 50

224 Kosambi, D., "Statistics in Function Space," *Journal of Indian Mathematical Society*, Vol. 7, 1943, pp. 76–88. DOI: 10.18311/jims/1943/17156 cited on p. 50

225 Mees, A. I., Rapp, P., and Jennings, L., "Singular-value decomposition and embedding dimension," *Physical Review A*, Vol. 36, 1 1987, pp. 340–346. DOI: 10.1103/PhysRevA.36.340 cited on p. 50

226 Liang, Y., Lee, H., Lim, S., Lin, W., Lee, K., and Wu, C., "Proper orthogonal decomposition and its applications–Part I: Theory," *Journal of Sound and Vibration*, Vol. 252, No. 3, 2002, pp. 527–544. DOI: 10.1006/jsvi.2001.4041 cited on p. 50

227 Wu, C., Liang, Y., Lin, W., Lee, H., and Lim, S., "A note on equivalence of proper orthogonal decomposition methods," *Journal of Sound and Vibration*, Vol. 265, No. 5, 2003, pp. 1103–1110. DOI: 10.1016/S0022-460X(03)00032-4 cited on p. 50

228 Kerschen, G. and Golinval, J.-C., "Physical interpretation of the proper orthogonal modes using the singular value decomposition," *Journal of Sound and Vibration*, Vol. 249, No. 5, 2002, pp. 849–865.
DOI: 10.1006/jsvi.2001.3930 cited on p. 50

229 Lenaerts, V., Kerschen, G., and Golinval, J.-C., "Identification of a continuous structure with a geometrical non-linearity. Part II: Proper orthogonal decomposition," *Journal of Sound and Vibration*, Vol. 262, No. 4, 2003, pp. 907–919.
DOI: 10.1016/S0022-460X(02)01132-X cited on p. 50

230 Gobat, G., Opreni, A., Fresca, S., Manzoni, A., and Frangi, A., "Reduced order modeling of nonlinear microstructures through Proper Orthogonal Decomposition," *Mechanical Systems and Signal Processing*, Vol. 171, 2022, p. 108 864.
DOI: 10.1016/j.ymssp.2022.108864 cited on p. 50

231 Corigliano, A., Dossi, M., and Mariani, S., "Domain decomposition and model order reduction methods applied to the simulation of multi-physics problems in MEMS," *Computers & Structures*, Vol. 122, 2013, pp. 113–127.
DOI: 10.1016/j.compstruc.2012.12.012 cited on p. 50

232 Corigliano, A., Dossi, M., and Mariani, S., "Model Order Reduction and domain decomposition strategies for the solution of the dynamic elastic-plastic structural problem," *Computer Methods in Applied Mechanics and Engineering*, Vol. 290, 2015, pp. 127–155, ISSN: 0045-7825.
DOI: 10.1016/j.cma.2015.02.021 cited on p. 50

233 Guo, M. and Hesthaven, J. S., "Reduced order modeling for nonlinear structural analysis using Gaussian process regression," *Computer Methods in Applied Mechanics and Engineering*, Vol. 341, 2018, pp. 807–826.
DOI: 10.1016/j.cma.2018.07.017 cited on p. 50

234 Tezzele, M., Fabris, L., Sidari, M., Sicchiero, M., and Rozza, G., "A multi-fidelity approach coupling parameter space reduction and non-intrusive POD with application to structural optimization of passenger ship hulls," *International Journal for Numerical Methods in Engineering*, Vol. 124, No. 5, 2023, pp. 1193–1210.
DOI: 10.1002/nme.7159 cited on p. 50

235 Corigliano, A., Confalonieri, F., Dossi, M., and Mariani, S., "Coupled domain decomposition-proper orthogonal decomposition methods for the simulation of quasi-brittle fracture processes," *Advanced Modeling and Simulation in Engineering Sciences*, Vol. 3, No. 1, 2016, pp. 1–24.
DOI: 10.1186/s40323-016-0081-9 cited on p. 50

236 Eckart, C. and Young, G., "The approximation of one matrix by another of lower rank," *Psychometrika*, Vol. 1, 3 1936, pp. 211–218. DOI: 10.1007/BF02288367 cited on pp. 51, 83

237 Feeny, B. and Kappagantu, R., "On the physical interpretation of proper orthogonal modes in vibrations," *Journal of Sound and Vibration*, Vol. 211, No. 4, 1998, pp. 607–616. DOI: 10.1006/jsvi.1997.1386 cited on p. 53

238 Azam, S. E. and Mariani, S., "Investigation of computational and accuracy issues in POD-based reduced order modeling of dynamic structural systems," *Engineering Structures*, Vol. 54, 2013, pp. 150–167. DOI: 10.1016/j.engstruct.2013.04.004 cited on p. 54

239 Torzoni, M., Tezzele, M., Mariani, S., Manzoni, A., and Willcox, K. E., "A digital twin framework for civil engineering structures," *Computer Methods in Applied Mechanics and Engineering*, Vol. 418, 2024, p. 116 584. DOI: 10.1016/j.cma.2023.116584 cited on pp. 59, 84, 185

240 Easmussen, C. E., "Gaussian processes in machine learning," *Advanced Lectures on Machine Learning*, Bousquet, O., Luxburg, U. von, and Rätsch, G., Eds. Berlin, Germany: Springer, 2004, pp. 63–71. DOI: 10.1007/978-3-540-28650-9_4 cited on pp. 61, 63

241 Quinlan, J. R., "Induction of decision trees," *Machine Learning*, Vol. 1, 1986, pp. 81–106. DOI: 10.1007/BF00116251 cited on p. 63

242 Beck, J. L. and Yuen, K.-V., "Model Selection Using Response Measurements: Bayesian Probabilistic Approach," *Journal of Engineering Mechanics*, Vol. 130, No. 2, 2004, pp. 192–203. DOI: 10.1061/(ASCE)0733-9399(2004)130:2(192) cited on p. 64

243 Hoerl, A. E. and Kennard, R. W., "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, Vol. 12, No. 1, 1970, pp. 55–67. DOI: 10.2307/1267351 cited on p. 64

244 Rumelhart, D. E. and McClelland, J. L., "Learning internal representations by error propagation," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. MIT press, 1987, pp. 318–362. cited on pp. 65, 66, 69

245 Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back-propagating errors," *Nature*, Vol. 323, 6088 1986, pp. 533–536. DOI: 10.1038/323533a0 cited on pp. 65, 66, 69

246 McCulloch, W. S. and Pitts, W., "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, Vol. 5, 4 1943, pp. 115–133.
DOI: 10.1007/BF02478259 cited on p. 65

247 Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, Vol. 65, No. 6, 1958, p. 386.
DOI: 10.1037/h0042519 cited on p. 65

248 Cybenko, G., "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, Vol. 2, No. 4, 1989, pp. 303–314.
DOI: 10.1007/BF02551274 cited on p. 66

249 Chen, T. and Chen, H., "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems," *IEEE Transactions on Neural Networks*, Vol. 6, No. 4, 1995, pp. 911–917.
DOI: 10.1109/72.392253 cited on p. 66

250 Chollet, F. *et al.*, *Keras*, https://keras.io, 2015. cited on pp. 66, 105, 196

251 Géron, A., *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., 2019.
DOI: 10.5555/3378999 cited on pp. 69, 71

252 Chollet, F., *Deep Learning with Python*. Shelter Island, Ney York: Manning Publications, 2017. cited on pp. 69, 71

253 Glorot, X. and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," *Proceedings of the 30th International Conference on Artificial Intelligence and Statistics*, Vol. 9, ser. Proceedings of Machine Learning Research, Sardinia, Italy, 2010, pp. 249–256. cited on pp. 70, 263, 266, 272, 273

254 He, K., Zhang, X., Ren, S., and Sun, J., "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *2015 IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.
DOI: 10.1109/ICCV.2015.123 cited on pp. 70, 265

255 Kingma, D. and Ba, J., "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations*, Vol. 3, 2015, pp. 1–13. cited on pp. 71, 264, 265, 272, 273

256 Ioffe, S. and Szegedy, C., "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proceedings of the 32th International Conference on Machine Learning*, 2015, pp. 448–456.
DOI: 10.5555/3045118.3045167 cited on p. 72

257 Bergstra, J. and Bengio, Y., "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, Vol. 13, No. 10, 2012, pp. 281–305.
DOI: 10.5555/2188385.2188395 cited on p. 72

258 Wu, J., Chen, X.-Y., Zhang, H., Xiong, L.-D., Lei, H., and Deng, S.-H., "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization," *Journal of Electronic Science and Technology*, Vol. 17, No. 1, 2019, pp. 26–40.
DOI: 10.11989/JEST.1674-862X.80904120 cited on p. 72

259 Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A., "Hyperband: A novel bandit-based approach to hyperparameter optimization," *Journal of Machine Learning Research*, Vol. 18, No. 1, 2017, pp. 6765–6816.
DOI: 10.5555/3122009.3242042 cited on p. 72

260 Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A., "Extracting and composing robust features with denoising autoencoders," *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland: Association for Computing Machinery, 2008, pp. 1096–1103.
DOI: 10.1145/1390156.1390294 cited on p. 74

261 D'Alessandro, A., Vitale, G., Scudero, S., D'Anna, R., Costanza, A., Fagiolini, A., and Greco, L., "Characterization of MEMS accelerometer self-noise by means of PSD and Allan Variance analysis," *7th IEEE International Workshop on Advances in Sensors and Interfaces*, 2017, pp. 159–164.
DOI: 10.1109/IWASI.2017.7974238 cited on p. 74

262 Evans, J. R., Allen, R. M., Chung, A. I., Cochran, E. S., Guy, R., Hellweg, M., and Lawrence, J. F., "Performance of Several Low-Cost Accelerometers," *Seismological Research Letters*, Vol. 85, No. 1, 2014, pp. 147–158.
DOI: 10.1785/0220130091 cited on p. 74

263 Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, Vol. 15, No. 56, 2014, pp. 1929–1958.
DOI: 10.5555/2627435.2670313 cited on p. 74

264 He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.
DOI: 10.1109/CVPR.2016.90 cited on pp. 75, 265

265 Jastrzębski, S., Arpit, D., Ballas, N., Verma, V., Che, T., Bengio, Y. J., Arpit, D., Ballas, N., Verma, V., Che, T., and Bengio, Y., "Residual Connections Encourage Iterative Inference," *6th International Conference on Learning Representations*, 2018. cited on p. 75

266 Bello, I., Fedus, I. B., Fedus, W., Du, X. D., Cubuk, E. D., Srinivas, A., Lin, T.-Y., Shlens, J., Zoph, C., Srinivas, A., Lin, T.-Y., Shlens, J., and Zoph, B., "Revisiting ResNets: Improved Training and Scaling Strategies," *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2021, pp. 22 614–22 627. cited on p. 75

267 Veit, A., Wilber, M., and Belongie, S., "Residual networks behave like ensembles of relatively shallow networks," *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Red Hook, New York: Curran Associates Inc., 2016. DOI: 10.5555/3157096.3157158 cited on p. 75

268 Hubel, D. H., "Single unit activity in striate cortex of unrestrained cat," *The Journal of physiology*, Vol. 147, No. 2, 1959, pp. 226–238. DOI: 10.1113/jphysiol.1959.sp006238 cited on p. 75

269 LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L., "Handwritten Digit Recognition with a Back-Propagation Network," *Proceedings of the 2nd International Conference on Neural Information Processing Systems*, Touretzky, D., Ed., Morgan-Kaufmann, 1989. cited on p. 75

270 Krizhevsky, A., Sutskever, I., and Hinton, G. E., "ImageNet Classification with Deep Convolutional Neural Networks," *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Pereira, F., Burges, C., Bottou, L., and Weinberger, K., Eds., Vol. 25, Curran Associates Inc., 2012. cited on p. 75

271 LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Single unit activity in striate cortex of unrestrained cats," *Proceedings of the IEEE*, Vol. 86, No. 11, 1959, pp. 2278–2324. DOI: 10.1109/5.726791 cited on p. 75

272 Avci, O., Abdeljaber, O., Kiranyaz, S., Hussein, M., Gabbouj, M., and Inman, D. J., "A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications," *Mechanical Systems and Signal Processing*, Vol. 147, 2021, p. 107 077. DOI: 10.1016/j.ymssp.2020.107077 cited on p. 76

273 Hu, J., Shen, L., and Sun, G., "Squeeze–and–Excitation Networks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, 2018, pp. 7132–7141. DOI: 10.1109/CVPR.2018.00745 cited on p. 77

274 Battaglia, P., Hamrick, J. B. C., Bapst, V., Sanchez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K., Nash, C., Langston, V. J., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R., "Relational inductive biases, deep learning, and graph networks," arXiv preprint arXiv:1806.01261, 2018 cited on p. 77

275 Wang, Z., Yan, W., and Oates, T., "Time series classification from scratch with deep neural networks: A strong baseline," *2017 International Joint Conference on Neural Networks*, Anchorage, Alaska, 2017, pp. 1578–1585.
DOI: 10.1109/IJCNN.2017.7966039 cited on pp. 77, 263, 273

276 Zhang, T., Biswal, S., and Wang, Y., "SHMnet: Condition assessment of bolted connection with beyond human-level performance," *Structural Health Monitoring*, Vol. 19, No. 4, 2020, pp. 1188–1201.
DOI: 10.1177/1475921719881237 cited on p. 77

277 Ewald, V., Groves, R. M., and Benedictus, R., "DeepSHM: A deep learning approach for structural health monitoring based on guided Lamb wave technique," *Proceeding of SPIE Smart Structures/NDE 2019*, 2019.
DOI: 10.1117/12.2506794 cited on p. 77

278 Giglioni, V., Venanzi, I., Poggioni, V., Milani, A., and Ubertini, F., "Autoencoders for unsupervised real-time bridge health assessment," *Computer-Aided Civil and Infrastructure Engineering*, Vol. 38, 2022, pp. 959–974.
DOI: 10.1111/mice.12943 cited on pp. 77, 81

279 Zhou and Chellappa, "Computation of optical flow using a neural network," *1988 IEEE International Conference on Neural Networks*, 1988, pp. 71–78.
DOI: 10.1109/ICNN.1988.23914 cited on p. 77

280 Zhang, R., Liu, Y., and Sun, H., "Physics-informed multi-LSTM networks for metamodeling of nonlinear structures," *Computer Methods in Applied Mechanics and Engineering*, Vol. 369, 2020, p. 113 226.
DOI: 10.1016/j.cma.2020.113226 cited on p. 78

281 Zhang, R., Chen, Z., Chen, S., Zheng, J., Büyüköztürk, O., and Sun, H., "Deep long short-term memory networks for nonlinear structural seismic response prediction," *Computers & Structures*, Vol. 220, 2019, pp. 55–68.
DOI: 10.1016/j.compstruc.2019.05.006 cited on p. 78

282 Simpson, T., Dervilis, N., and Chatzi, E., "Machine Learning Approach to Model Order Reduction of Nonlinear Systems via Autoencoder and LSTM Networks," *Journal of Engineering Mechanics*, Vol. 147, No. 10, 2021, p. 04 021 061. DOI: 10.1061/(ASCE)EM.1943-7889.0001971 cited on pp. 78, 81

283 Hochreiter, S. and Schmidhuber, J., "Long short-term memory," *Neural Computation*, Vol. 9, No. 8, 1997, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735 cited on p. 79

284 Olah, C., *Understanding LSTM Networks*, https://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015. cited on p. 79

285 Kramer, M. A., "Nonlinear principal component analysis using autoassociative neural networks," *American Institute of Chemical Engineers Journal*, Vol. 37, No. 2, 1991, pp. 233–243. DOI: 10.1002/aic.690370209 cited on p. 81

286 Kerschen, G. and Golinval, J.-C., "Feature extraction using auto-associative neural networks," *Smart Materials and Structures*, Vol. 13, No. 1, 2003, p. 211. DOI: 10.1088/0964-1726/13/1/025 cited on p. 81

287 Hinton, G. E. and Salakhutdinov, R. R., "Reducing the Dimensionality of Data with Neural Networks," *Science*, Vol. 313, No. 5786, 2006, pp. 504–507. DOI: 10.1126/science.1127647 cited on p. 81

288 Hinton, G. E., Krizhevsky, A., and Wang, S. D., "Transforming auto-encoders," *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks*, Espoo, Finland: Springer, 2011, pp. 44–51. cited on p. 81

289 Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., and Bottou, L., "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion," *Journal of Machine Learning Research*, Vol. 11, No. 110, 2010, pp. 3371–3408. cited on p. 81

290 Liou, C.-Y., Cheng, W.-C., Liou, J.-W., and Liou, D.-R., "Autoencoder for words," *Neurocomputing*, Vol. 139, 2014, pp. 84–96. DOI: 10.1016/j.neucom.2013.09.055 cited on p. 81

291 Japkowicz, N., Hanson, S. J., and Gluck, M. A., "Nonlinear autoassociation is not equivalent to PCA," *Neural Computation*, Vol. 12, No. 3, 2000, pp. 531–545. DOI: 10.1162/089976600300015691 cited on p. 81

292 Kingma, D. P., Welling, M., *et al.*, "An Introduction to Variational Autoencoders," *Foundations and Trends® in Machine Learning*, Vol. 12, No. 4, 2019, pp. 307–392, ISSN: 1935-8237. DOI: 10.1561/2200000056 cited on p. 81

293 Salakhutdinov, R. and Hinton, G., "Semantic hashing," *International Journal of Approximate Reasoning*, Vol. 50, No. 7, 2009, pp. 969–978, ISSN: 0888-613X.
DOI: 10.1016/j.ijar.2008.11.006 cited on p. 81

294 Kingma, D. P. and Welling, M., "Auto-Encoding Variational Bayes," arXiv preprint arXiv:1312.6114, 2013 cited on p. 81

295 Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B., "Adversarial Autoencoders," arXiv preprint arXiv:1511.05644, 2015 cited on p. 81

296 Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B., "Wasserstein Auto-Encoders," arXiv preprint arXiv:1711.01558, 2017 cited on p. 81

297 Sohn, K., Lee, H., and Yan, X., "Learning Structured Output Representation using Deep Conditional Generative Models," *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., Eds., Curran Associates, Inc., 2015. cited on p. 81

298 Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A., "Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework," *Proceedings of the 5th International Conference on Learning Representations*, 2017. cited on p. 81

299 Sohn, H., Worden, K., and Farrar, C. R., "Novelty detection under changing environmental conditions," *Smart Structures and Materials 2001: Smart Systems for Bridges, Structures, and Highways*, Vol. 4330, 2001, pp. 108–118.
DOI: 10.1117/12.434110 cited on p. 81

300 Rosafalco, L., Manzoni, A., Mariani, S., and Corigliano, A., "An Autoencoder-Based Deep Learning Approach for Load Identification in Structural Dynamics," *Sensors*, Vol. 21, No. 12, 2021, p. 4207.
DOI: {10.3390/s21124207} cited on p. 81

301 Ma, X., Lin, Y., Nie, Z., and Ma, H., "Structural damage identification based on unsupervised feature-extraction via Variational Auto-encoder," *Measurement*, Vol. 160, 2020, p. 107 811.
DOI: 10.1016/j.measurement.2020.107811 cited on p. 81

302 Worden, K., Manson, G., and Allman, D., "Experimental validation of a structural health monitoring methodology: Part I. Novelty detection on a laboratory structure," *Journal of Sound and Vibration*, Vol. 259, No. 2, 2003, pp. 323–343.
DOI: https://doi.org/10.1006/jsvi.2002.5168 cited on p. 81

303 Pathirage, C. S. N., Li, J., Li, L., Hao, H., Liu, W., and Ni, P., "Structural damage identification based on autoencoder neural networks and deep learning," *Engineering Structures*, Vol. 172, 2018, pp. 13–28.
DOI: 10.1016/j.engstruct.2018.05.109 cited on p. 81

304 Simpson, T., Dervilis, N., Couturier, P., Maljaars, N., and Chatzi, E., "Reduced order modeling of non-linear monopile dynamics via an AE-LSTM scheme," *Frontiers in Energy Research*, Vol. 11, 2023, p. 1 128 201.
DOI: 10.3389/fenrg.2023.1128201 cited on p. 81

305 Liu, W., Lai, Z., Bacsa, K., and Chatzi, E., "Physics-guided Deep Markov Models for learning nonlinear dynamical systems with uncertainty," *Mechanical Systems and Signal Processing*, Vol. 178, 2022, p. 109 276.
DOI: 10.1016/j.ymssp.2022.109276 cited on p. 81

306 Mylonas, C., Abdallah, I., and Chatzi, E., "Conditional variational autoencoders for probabilistic wind turbine blade fatigue estimation using Supervisory, Control, and Data Acquisition data," *Wind Energy*, Vol. 24, No. 10, 2021, pp. 1122–1139.
DOI: 10.1002/we.2621 cited on p. 81

307 Giglioni, V., Venanzi, I., Baia, A. E., Poggioni, V., Milani, A., and Ubertini, F., *Deep autoencoders for unsupervised damage detection with application to the z24 benchmark bridge*, *European Workshop on Structural Health Monitoring*, Springer International Publishing, 2023, pp. 1048–1057.
DOI: 10.1007/978-3-031-07258-1_105 cited on pp. 83, 168

308 Garcia, G. R., Michau, G., Ducoffe, M., Gupta, J. S., and Fink, O., "Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, Vol. 236, No. 4, 2022, pp. 617–627.
DOI: 10.1177/1748006X21994446 cited on pp. 83, 168, 270

309 Mantawy, I. M. and Mantawy, M. O., "Convolutional neural network based structural health monitoring for rocking bridge system by encoding time-series into images," *Structural Control and Health Monitoring*, Vol. 29, No. 3, 2022, e2897.
DOI: 10.1002/stc.2897 cited on pp. 83, 168

310 Lopez-Paz, D. and Sagun, L., "Easing non-convex optimization with neural networks," *Proceedings of the 6th International Conference on Learning Representations*, 2018. cited on p. 83

311 Rolinek, M., Zietlow, D., and Martius, G., "Variational Autoencoders Pursue PCA Directions (by Accident)," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 398–12 407.
DOI: 10.1109/CVPR.2019.01269 cited on p. 83

312 Sucar, L. E., *Probabilistic Graphical Models: Principles and Applications*. Springer London.
DOI: 10.1007/978-1-4471-6699-3 cited on p. 83

313 Warner, H. R., Toronto, A. F., Veasey, L. G., and Stephenson, R., "A Mathematical Approach to Medical Diagnosis: Application to Congenital Heart Disease," *Journal of the American Medical Association*, Vol. 177, No. 3, 1961, pp. 177–183.
DOI: 10.1001/jama.1961.03040290005002 cited on p. 84

314 Pearl, J., *Probabilistic Reasoning in Intelligent Systems*. San Francisco, California: Morgan Kaufmann, 1988.
DOI: 10.1016/B978-0-08-051489-5.50007 cited on p. 84

315 Andriotis, C. P. and Papakonstantinou, K. G., "Managing engineering systems with large state and action spaces through deep reinforcement learning," *Reliability Engineering & System Safety*, Vol. 191, 2019, p. 106 483.
DOI: 10.1016/j.ress.2019.04.036 cited on p. 84

316 Morato, P. G., Andriotis, C. P., Papakonstantinou, K. G., and Rigo, P., "Inference and dynamic decision-making for deteriorating systems with probabilistic dependencies through Bayesian networks and deep reinforcement learning," *Reliability Engineering & System Safety*, Vol. 235, 2023, p. 109 144.
DOI: 10.1016/j.ress.2023.109144 cited on p. 84

317 Papakonstantinou, K. and Shinozuka, M., "Optimum inspection and maintenance policies for corroded structures using partially observable Markov decision processes and stochastic, physically based models," *Probabilistic Engineering Mechanics*, Vol. 37, 2014, pp. 93–108.
DOI: 10.1016/j.probengmech.2014.06.002 cited on p. 84

318 Papakonstantinou, K. G. and Shinozuka, M., "Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part I: Theory," *Reliability Engineering & System Safety*, Vol. 130, 2014, pp. 202–213.
DOI: 10.1016/j.ress.2014.04.005 cited on pp. 84, 91

319 Papakonstantinou, K. G. and Shinozuka, M., "Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation," *Reliability Engineering & System Safety*, Vol. 130, 2014, pp. 214–224.
DOI: 10.1016/j.ress.2014.04.006 cited on pp. 84, 193, 201

320 Hlaing, N., Morato, P. G., Nielsen, J. S., Amirafshari, P., Kolios, A., and Rigo, P., "Inspection and maintenance planning for offshore wind structural components: integrating fatigue failure criteria with Bayesian networks and Markov decision processes," *Structure and Infrastructure Engineering*, Vol. 18, No. 7, 2022, pp. 983–1001.
DOI: 10.1080/15732479.2022.2037667 cited on p. 84

321 Hughes, A. J., Barthorpe, R. J., Dervilis, N., Farrar, C. R., and Worden, K., "A probabilistic risk-based decision framework for structural health monitoring," *Mechanical Systems and Signal Processing*, Vol. 150, 2021, p. 107 339.
DOI: 10.1016/j.ymssp.2020.107339 cited on p. 84

322 Morato, P. G., Papakonstantinou, K. G., Andriotis, C. P., Nielsen, J. S., and Rigo, P., "Optimal inspection and maintenance planning for deteriorating structural components through dynamic Bayesian networks and Markov decision processes," *Structural Safety*, Vol. 94, 2022, p. 102 140.
DOI: 10.1016/j.strusafe.2021.102140 cited on p. 84

323 Andriotis, C. P. and Papakonstantinou, K. G., "Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints," *Reliability Engineering & System Safety*, Vol. 212, 2021, p. 107 551.
DOI: 10.1016/j.ress.2021.107551 cited on p. 84

324 Karve, P. M., Guo, Y., Kapusuzoglu, B., Mahadevan, S., and Haile, M. A., "Digital twin approach for damage-tolerant mission planning under uncertainty," *Engineering Fracture Mechanics*, Vol. 225, 2020, p. 106 766.
DOI: 10.1016/j.engfracmech.2019.106766 cited on p. 84

325 Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press, 2018. cited on p. 84

326 Pearl, J., "Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach," *Probabilistic and Causal Inference: The Works of Judea Pearl*, Geffner, H., Dechter, R., and Halpern, J. Y., Eds., 1982, pp. 129–138. cited on pp. 87, 191

327 Bellman, R., "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, Vol. 6, No. 5, 1957, pp. 679–684. cited on p. 91

328 Howard, R. A., *Dynamic programming and Markov processes*. Oxford, England: John Wiley, 1960. cited on p. 91

329 Bellman, R., *Dynamic Programming*. Princeton, New Jersey: Princeton University Press, 1957. cited on p. 93

330 Negri, F., *RedbKIT, version 2.2*, http://redbkit.github.io/redbKIT, 2016. cited on pp. 103, 194

331 Salakhutdinov, R. and Hinton, G., "Learning a Nonlinear Embedding by Preserving Class Neighborhood Structure," *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, Vol. 2, San Juan, Puerto Rico, 2007, pp. 412–419. cited on p. 103

332 LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F., "A tutorial on energy-based learning," *Predicting structured data*, MIT Press, 2006. cited on pp. 105, 106

333 Liu, C.-L. and Chen, Q.-H., "Metric-Based Semi-Supervised Regression," *IEEE Access*, Vol. 8, 2020, pp. 30 001–30 011. DOI: 10.1109/ACCESS.2020.2971229 cited on p. 107

334 Hastie, T., Tibshirani, R., and Friedman, J., *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, New York: Springer., 2009. DOI: 10.1007/978-0-387-84858-7 cited on p. 113

335 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.*, "Scikit-learn: Machine Learning in Python," *The Journal of machine Learning research*, Vol. 12, 2011, pp. 2825–2830. cited on p. 113

336 Arvidsson, T. and Li, J., "Dynamic analysis of a portal frame railway bridge using frequency dependent soil structure interaction," Master thesis, KTH Royal Institute of Technology, 2011. cited on p. 122

337 Ülker-Kaustell, M., "Some aspects of the dynamic soil-structure interaction of a portal frame railway bridge," Ph.D. Dissertation, KTH Royal Institute of Technology, 2009. cited on p. 122

338 Zakić, B., Ryzynski, A., Guo-Hong, C., and Jokela, J., "Classification of damage in concrete bridges," *Materials and Structures*, Vol. 24, 4 1991, pp. 268–275. DOI: 10.1007/BF02472082 cited on p. 122

339 European Committee for Standardization, "Part 2: Traffic loads on bridges," *EN 1991-2 Eurocode 1: Actions on structures*, 2003, pp. 66–74. cited on p. 124

340 Torzoni, M., Manzoni, A., and Mariani, S., "A Multi-Fidelity Deep Neural Network Approach to Structural Health Monitoring," *Engineering Proceedings*, Vol. 27, No. 1, 2022. DOI: 10.3390/ecsa-9-13344 cited on p. 135

341 Ritter, A. and Muñoz-Carpena, R., "Performance evaluation of hydrological models: Statistical significance for reducing subjectivity in goodness-of-fit assessments," *Journal of Hydrology*, Vol. 480, 2013, pp. 33–45. DOI: 10.1016/j.jhydrol.2012.12.004 cited on p. 144

342 Papadimitriou, C. and Lombaert, G., "The effect of prediction error correlation on optimal sensor placement in structural dynamics," *Mechanical Systems and Signal Processing*, Vol. 28, 2012, pp. 105–127.
DOI: 10.1016/j.ymssp.2011.05.019 cited on p. 158

343 Capellari, G., Chatzi, E., and Mariani, S., "Structural Health Monitoring Sensor Network Optimization through Bayesian Experimental Design," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, Vol. 4, No. 2, 2018, p. 04 018 016.
DOI: 10.1061/AJRUA6.0000966 cited on p. 163

344 Bigoni, C., Zhang, Z., and Hesthaven, J. S., "Systematic sensor placement for structural anomaly detection in the absence of damaged states," *Computer Methods in Applied Mechanics and Engineering*, Vol. 371, 2020, p. 113 315.
DOI: 10.1016/j.cma.2020.113315 cited on p. 163

345 Marwan, N., Romano, M. C., Thiel, M., and Kurths, J., "Recurrence plots for the analysis of complex systems," *Physics Reports*, Vol. 438, No. 5, 2007, pp. 237–329.
DOI: 10.1016/j.physrep.2006.11.001 cited on p. 168

346 Xu, G., Liu, M., Jiang, Z., Shen, W., and Huang, C., "Online Fault Diagnosis Method Based on Transfer Convolutional Neural Networks," *IEEE Transactions on Instrumentation and Measurement*, Vol. 69, No. 2, 2020, pp. 509–520.
DOI: 10.1109/TIM.2019.2902003 cited on p. 168

347 Torzoni, M. and Tezzele, M., *Digital-twin-SHM*, https://github.com/mtezzele/digital-twin-SHM, 2023. cited on p. 185

348 Ferrari, A. and Willcox, K., "Digital twins in mechanical and aerospace engineering," *Nature Computational Science*, Vol. 4, 2024, pp. 178–183.
DOI: 10.1038/s43588-024-00613-8 cited on p. 188

349 Morcous, G., "Performance Prediction of Bridge Deck Systems Using Markov Chains," *Journal of Performance of Constructed Facilities*, Vol. 20, No. 2, 2006, pp. 146–155.
DOI: 10.1061/(ASCE)0887-3828(2006)20:2(146) cited on p. 191

350 Thompson, P. D. and Johnson, M. B., "Markovian bridge deterioration: developing models from historical data," *Structure and Infrastructure Engineering*, Vol. 1, No. 1, 2005, pp. 85–91.
DOI: 10.1080/15732470412331289332 cited on p. 191

351 Bianchi, S. and Biondini, F., "Life-Cycle Assessment of Deteriorating RC Bridges Using Artificial Neural Networks," *Journal of Infrastructure Systems*, Vol. 28, No. 2, 2022.
DOI: 10.1061/(ASCE)IS.1943-555X.0000672 cited on p. 191

352 Biondini, F. and Frangopol, D. M., "Time-variant redundancy and failure times of deteriorating concrete structures considering multiple limit states," *Structure and Infrastructure Engineering*, Vol. 13, No. 1, 2017, pp. 94–106. DOI: 10.1080/15732479.2016.1198403 cited on p. 191

353 Ankan, A. and Panda, A., "Pgmpy: Probabilistic graphical models using Python," *Proceedings of the 14th Python in Science Conference*, Citeseer, 2015. cited on p. 193

354 Biondini, F. and Garavaglia, E., "Probabilistic service life prediction and maintenance planning of deteriorating structures," *2005 International Conference on Structural Safety and Reliability*, 2005, pp. 19–22. cited on p. 218

355 Biondini, F., Bontempi, F., Frangopol, D. M., and Malerba, P. G., "Probabilistic Service Life Assessment and Maintenance Planning of Concrete Structures," *Journal of Structural Engineering*, Vol. 132, No. 5, 2006, pp. 810–825. DOI: 10.1061/(ASCE)0733-9445(2006)132:5(810) cited on p. 218

356 Rainforth, T., Foster, A., Ivanova, D. R., and Bickford Smith, F., "Modern Bayesian Experimental Design," *Statistical Science*, Vol. 39, No. 1, 2024, pp. 100–114. DOI: 10.1214/23-STS915 cited on p. 219

357 Cranmer, K., Brehmer, J., and Louppe, G., "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, Vol. 117, No. 48, 2020, pp. 30 055–30 062. DOI: 10.1073/pnas.1912789117 cited on p. 220

358 Karim, F., Majumdar, S., Darabi, H., and Harford, S., "Multivariate LSTM-FCNs for time series classification," *Neural Networks*, Vol. 116, 2019, pp. 237–245. DOI: 10.1016/j.neunet.2019.04.014 cited on pp. 263, 273

359 Donner, R. V., Zou, Y., Donges, J. F., Marwan, N., and Kurths, J., "Recurrence networks—a novel paradigm for nonlinear time series analysis," *New Journal of Physics*, Vol. 12, No. 3, 2010, p. 033 025. DOI: 10.1088/1367-2630/12/3/033025 cited on p. 269

360 Campanharo, A. S., Sirer, M. I., Malmgren, R. D., Ramos, F. M., and Amaral, L. A. N., "Duality between Time Series and Networks," *PLoS One*, Vol. 6, No. 8, 2011, pp. 1–13. DOI: 10.1371/journal.pone.0023378 cited on p. 269

361 Lin, J., Keogh, E., Wei, L., and Lonardi, S., "Experiencing SAX: a novel symbolic representation of time series," *Data Mining and Knowledge Discovery*, Vol. 15, 2 2007, pp. 107–144. DOI: 10.1007/s10618-007-0064-z cited on p. 270

This page intentionally left blank.

APPENDICES

This page intentionally left blank.

A IMPLEMENTATION DETAILS OF $\mathrm{NN}_{\mathbf{h}}$

In this Appendix, we discuss the implementation details of the DL models employed in Chapter 5. The architectures, as well as the relevant hyperparameters and training options, have been chosen through a preliminary study aimed at minimizing $\mathcal{L}_{\mathbf{h}}^{(1,2,3)}$ while retaining the generalization capabilities of $\mathrm{NN}_{\mathbf{h}}$.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathbf{h}}, L, N_u)$	None	–
1 - Conv1D	$(B_{\mathbf{h}}, L, 16)$	Tanh	0
2 - BatchNorm	$(B_{\mathbf{h}}, L, 16)$	None	1
3 - MaxPool1D	$(B_{\mathbf{h}}, L/2, 16)$	None	2
4 - Dropout	$(B_{\mathbf{h}}, L/2, 16)$	None	3
5 - Conv1D	$(B_{\mathbf{h}}, L/2, 32)$	Tanh	4
6 - BatchNorm	$(B_{\mathbf{h}}, L/2, 32)$	None	5
7 - MaxPool1D	$(B_{\mathbf{h}}, L/4, 32)$	None	6
8 - Dropout	$(B_{\mathbf{h}}, L/4, 32)$	None	7
9 - Conv1D	$(B_{\mathbf{h}}, L/4, 16)$	Tanh	8
10 - BatchNorm	$(B_{\mathbf{h}}, L/4, 16)$	None	9
11 - MaxPool1D	$(B_{\mathbf{h}}, L/8, 16)$	None	10
12 - Dropout	$(B_{\mathbf{h}}, L/8, 16)$	None	11
13 - Flatten	$(B_{\mathbf{h}}, 2L)$	None	12
14 - Dense	$(B_{\mathbf{h}}, 64)$	Tanh	13
15 - BatchNorm	$(B_{\mathbf{h}}, 64)$	None	14
16 - Dense	$(B_{\mathbf{h}}, 16)$	Tanh	15
17 - BatchNorm	$(B_{\mathbf{h}}, 16)$	None	16
18 - Dense	$(B_{\mathbf{h}}, N_h)$	None	17

(b)

Filter kernel size:	$H_C = \{25, 13, 7\}$
Pool size:	$H_{\mathcal{P}} = \{2, 2, 2\}$
Dropout rate:	5%
L^2 regularization rate:	$\lambda_{L^2} = \{10^{-3}, 10^{-4}\}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathbf{h}} = 32$
Initial learning rate:	$\eta_{\mathbf{\Omega_h}} = \{10^{-3}, 10^{-4}\}$
Allowed epochs:	100
Learning schedule:	0.8 cosine decay
Weight decay:	0.05
Early-stop patience:	15 epochs
Train-Val split:	80 : 20
Similarity threshold:	$\overline{\mathcal{E}_{\mathbf{y}}} = \{0.15, 0.25\}$
Positive pairings:	$N_+ = 10$
Negative pairings:	$N_- = 10$
Negative margin (Eucldiean):	$\overline{\epsilon_{\mathbf{h}}} = 1$
Negative margin (cosine):	$\overline{\epsilon_{\mathbf{h}}} = 0.4$

Table A.1: $\mathrm{NN}_{\mathbf{h}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

$\mathrm{NN}_{\mathbf{h}}$ is set as a 18-layers DL model, whose architecture is outlined in Tab. A.1a. At its root, $\mathrm{NN}_{\mathbf{h}}$ features a *damage-sensitive feature extractor* required to be insensitive to transformations in the input not related to damage. This is implemented through the composition of three 1D convolutional blocks, each provided with a Tanh-activated *1D convolutional layer*, *batch normalization*, *1D max pooling*, and *Dpout regularization*. The number of filter kernels is set to $N_F = \{16, 32, 16\}$, in accordance with a proportional scheme $\{N_F, 2N_F, N_F\}$, as suggested in [275], while the corresponding kernel sizes are set to $H_C = \{25, 13, 7\}$, following a scheme $H_C^{\ell} = H_C^{\ell-1}/2 + 1$ [358]. The extracted features are then reshaped through a *flatten layer* and run through a stack of three *fully-connected layers* featuring $\{64, 16, N_u\}$ neurons, respectively, and whose output is the low-dimensional representation $\mathbf{h}$: the first two are Tanh-activated before undergo *batch normalization*, while the output layer has no activation function.

275. Wang *et al.*, *Time series classification from scratch with deep neural networks: A strong baseline.* 2017

358. Karim *et al.*, *Multivariate LSTM-FCNs for time series classification.* 2019

Using *Xavier's weight initialization* [253], $\mathrm{NN}_{\mathbf{h}}$ is trained by minimizing the following *regularized loss function*:

253. Glorot *et al.*, *Understanding the difficulty of training deep feedforward neural networks.* 2010

$$\mathcal{L}_{\mathbf{h}}^{R2}(\mathbf{\Omega_h}, \mathcal{D}_{\mathrm{LOC}}^{\mathrm{P}}) = \mathcal{L}_{\mathbf{h}}(\mathbf{\Omega_h}, \mathcal{D}_{\mathrm{LOC}}^{\mathrm{P}}) + \lambda_{L^2} \|\mathtt{vec}[\mathbf{\Omega_h}]\|_2^2, \tag{A.1}$$

where $\lambda_{L^2} \in \mathbb{R}$ is the L^2 regularization rate over the model

parameters $\mathbf{\Omega_h}$. $\mathcal{L}_{\mathbf{h}}^{R2}$ is minimized using the first-order stochastic gradient descent optimizer *Adam* [255], for a max of 100 epochs.

255. Kingma *et al.*, *Adam: A Method for Stochastic Optimization*. 2015

The *learning rate* $\eta_{\mathbf{\Omega_h}}$ is initially set to $\{10^{-3}, 10^{-4}\}$, and decreased for 4/5 of the allowed training steps using a *cosine decay schedule* with decay equal to 0.05. The optimization is carried out using an 80 : 20 *splitting ratio* of the dataset for training and validation purposes, with 20% of the data randomly taken and set aside to monitor the learning process. We use an *early-stop strategy* to interrupt learning, whenever the loss function value attained on the validation set does not decrease for a prescribed number of patience epochs in a row. The hyperparameters and training options for $\mathrm{NN}_{\mathbf{h}}$ are reported in Tab. A.1b.

B IMPLEMENTATION DETAILS OF $\mathrm{NN}_{\mathrm{MF}}$ AND $\mathrm{NN}_{\mathrm{OC}}$

In this Appendix, we discuss the implementation details of the DL models employed in Chapter 6. The architectures and training options have been chosen through a preliminary study aimed at minimizing $\mathcal{L}_{\mathrm{LF}}$, $\mathcal{L}_{\mathrm{HF}}$, and $\mathcal{L}_{\mathrm{OC}}$ while retaining the generalization capabilities of $\mathrm{NN}_{\mathrm{LF}}$, $\mathrm{NN}_{\mathrm{HF}}$, and $\mathrm{NN}_{\mathrm{OC}}$.

$\mathrm{NN}_{\mathrm{LF}}$ is set as a 9-layers *fully-connected DL model*, whose architecture is outlined in Tab. B.1a. Each layer features 30 neurons, except for the last two layers, which have $2L_{\mathrm{LF}}$ and L_{LF} neurons, respectively. Two double-layer *residual connections* [264] are exploited to progressively refine the $\mathrm{NN}_{\mathrm{LF}}$ performance without introducing extra parameters. PReLU [254] is employed as the activation function in all the layers except the last, which has no activation.

264. He *et al.*, *Deep Residual Learning for Image Recognition*

254. He *et al.*, *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. 2015

Using *He's weight initialization* [254], $\mathrm{NN}_{\mathrm{LF}}$ is trained by minimizing the following *regularized loss function*:

$$\mathcal{L}_{\mathrm{LF}}^{R2}(\boldsymbol{\Omega}_{\mathrm{LF}}, \mathcal{D}_{\mathrm{LF}}^{\mathtt{vec}}) = \mathcal{L}_{\mathrm{LF}}(\boldsymbol{\Omega}_{\mathrm{LF}}, \mathcal{D}_{\mathrm{LF}}^{\mathtt{vec}}) + \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{LF}}]\|_2^2, \quad \text{(B.1)}$$

where $\lambda_{L^2} \in \mathbb{R}$ denotes the L^2 regularization rate over the model parameters $\boldsymbol{\Omega}_{\mathrm{LF}}$. $\mathcal{L}_{\mathrm{LF}}^{R2}$ is minimized using the first-order stochastic gradient descent optimizer *Adam* [255].

255. Kingma *et al.*, *Adam: A Method for Stochastic Optimization*. 2015

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{LF}}, N_{\mathrm{par}}^{\mathrm{LF}})$	None	–
1 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	0
2 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	1
3 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	2
4 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	3
5 - Add	$(B_{\mathrm{LF}}, 30)$	None	2,4
6 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	5
7 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	6
8 - Dense	$(B_{\mathrm{LF}}, 30)$	PReLU	7
9 - Add	$(B_{\mathrm{LF}}, 30)$	None	6,8
10 - Dense	$(B_{\mathrm{LF}}, 2L_{\mathrm{LF}})$	PReLU	9
11 - Dense	$(B_{\mathrm{LF}}, L_{\mathrm{LF}})$	None	10

(b)

L^2 regularization rate:	$\lambda_{L^2} = 10^{-6}$
Weight initializer:	He
Optimizer:	Adam
Batch size:	$B_{\mathrm{LF}} = 64$
Initial learning rate:	$\eta_{\boldsymbol{\Omega}_{\mathrm{LF}}} = 0.005$
Allowed epochs:	$10,000$
Learning schedule:	0.8 cosine decay
Weight decay:	0.02
Early-stop patience:	500 epochs
Train-Val split:	80 : 20

Table B.1: $\mathrm{NN}_{\mathrm{LF}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

The *learning rate* $\eta_{\boldsymbol{\Omega}_{\mathrm{LF}}}$ is initially set to 0.005 and then decreased for 4/5 of the allowed training steps using a *cosine decay schedule* with weight decay equal to 0.05. The optimization is carried out with an 80 : 20 *splitting ratio* of the dataset for training and validation, with 20% of the data randomly set aside to monitor the learning process. We employ an *early-stop strategy* to interrupt learning when the loss function value on the validation set does not decrease for a prescribed number of patience epochs in a row. The hyperparameters and training options for $\mathrm{NN}_{\mathrm{LF}}$ are detailed in Tab. B.1b.

$\mathrm{NN_{HF}}$ consists of four *LSTM layers* with cell states of size $\{16, 16, 32, N_u\}$, and a *time-distributed fully-connected output layer* with N_u neurons. The LSTM layers feature standard cells with Sigmoid gating functions and Tanh-activated cell states, while no activation is applied to the dense layer. The architecture of $\mathrm{NN_{HF}}$ is outlined in Tab. B.2a.

The optimization of $\mathrm{NN_{HF}}$ is carried out using *Xavier's weight initialization* [253] together with *Adam* optimizer to minimize the following *regularized loss function*:

253. Glorot *et al.*, *Understanding the difficulty of training deep feedforward neural networks*. 2010

$$\mathcal{L}^{R2}_{\mathrm{HF}}(\boldsymbol{\Omega}_{\mathrm{HF}}, \mathcal{D}_{\mathrm{HF}}) = \mathcal{L}_{\mathrm{HF}}(\boldsymbol{\Omega}_{\mathrm{HF}}, \mathcal{D}_{\mathrm{HF}}) + \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{HF}}]\|_2^2. \quad \text{(B.2)}$$

Similarly to what specified for $\mathrm{NN_{LF}}$, the *learning rate* $\eta_{\boldsymbol{\Omega}_{\mathrm{HF}}}$ is decreased using a *cosine decay schedule*. An *early-stop strategy* is adopted to prevent overfitting, with an 80 : 20 *splitting ratio* for training and validation. The relevant hyperparameters and the training options are summarized in Tab. B.2b.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{HF}}, L_{\mathrm{C}}, N^{\mathrm{HF}}_{\mathrm{par}} + N_u)$	None	–
1 - LSTM	$(B_{\mathrm{HF}}, L_{\mathrm{C}}, 16)$	Usual	0
2 - LSTM	$(B_{\mathrm{HF}}, L_{\mathrm{C}}, 16)$	Usual	1
3 - LSTM	$(B_{\mathrm{HF}}, L_{\mathrm{C}}, 32)$	Usual	2
4 - LSTM	$(B_{\mathrm{HF}}, L_{\mathrm{C}}, N_u)$	Usual	3
5 - Dense	(B_{HF}, N_u) - distributed	None	4

(b)

L^2 regularization rate:	$\lambda_{L^2} = \{10^{-3}, 10^{-4}\}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathrm{HF}} = 128$
Initial learning rate:	$\eta_{\boldsymbol{\Omega}_{\mathrm{HF}}} = 10^{-3}$
Allowed epochs:	$\{2000, 5000\}$
Learning schedule:	0.8 cosine decay
Weight decay:	0.05
Early-stop patience:	100 epochs
Chunks length:	$L_{\mathrm{C}} = 20$
Train-Val split:	80 : 20

Table B.2: $\mathrm{NN_{HF}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

$\mathrm{NN_{OC}}$ is set as a 2-layers *fully-connected DL model*. The hidden layer features 3 neurons and is equipped with Softsign activation, while the output layer features N_u neurons and has no activation.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{OC}}, L_{\mathrm{OC}})$	None	–
1 - Dense	$(B_{\mathrm{OC}}, 3)$	Softsign	0
2 - Dense	(B_{OC}, N_h)	None	1

(b)

L^2 regularization rate:	$\lambda_{L^2} = 0.0005$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathrm{OC}} = 32$
Initial learning rate:	$\eta_{\boldsymbol{\Omega}_{\mathrm{OC}}} = 10^{-4}$
Allowed epochs:	400
Learning schedule:	0.8 cosine decay
Weight decay:	0.01
Early-stop patience:	50 epochs
Train-Val split:	80 : 20
Positive pairings:	$N_+ = 10$
Negative pairings:	$N_- = 10$
Negative margin (cosine):	$\overline{\epsilon_{\mathbf{h}}} = 0.4$

Table B.3: $\mathrm{NN_{OC}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

Using *Xavier's weight initialization*, $\mathrm{NN_{OC}}$ is trained by minimizing the following *regularized loss function*:

$$\mathcal{L}^{R2}_{\mathrm{OC}}(\boldsymbol{\Omega}_{\mathrm{OC}}, \mathcal{D}^{\mathrm{P}}_{\mathrm{OC}}) = \mathcal{L}_{\mathrm{OC}}(\boldsymbol{\Omega}_{\mathrm{OC}}, \mathcal{D}^{\mathrm{P}}_{\mathrm{OC}}) + \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{OC}}]\|_2^2. \quad \text{(B.3)}$$

Also in this case, $\mathcal{L}_{\text{OC}}^{R2}$ is minimized using *Adam*, and the learning rate $\eta_{\boldsymbol{\Omega}_{\text{OC}}}$ is decreased as the training advances using a *cosine decay schedule*. *Early stopping* is also used to prevent overfitting using an 80 : 20 *splitting ratio* for training and validation. The architecture and training options of NN_{OC} are summarized in Tab. B.3a and in Tab. B.3b, respectively.

This page intentionally left blank.

C IMAGING TIME SERIES: MARKOV TRANSITION FIELD

In this Appendix, we review the *MTF encoding* [160] used in Chapter 7 to transform vibrations $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_{N_u}] \in \mathbb{R}^{L \times N_u}$ into images. The technique is described in detail with reference to a univariate time series, and it is applied identically to all N_u input channels.

160. Wang *et al.*, *Imaging Time-Series to Improve Classification and Imputation*. 2015

The MTF encoding can be traced back to the use of *recurrence networks* for analyzing the structural properties of time series. As proposed in [359], the *recurrence matrix* of a time series can be interpreted as the *adjacency matrix* of an associated *complex network*. In [360], the concept of building adjacency matrices has been extended by extracting *transition dynamics* from *first-order Markov matrices*, as described below.

359. Donner *et al.*, *Recurrence networks—a novel paradigm for nonlinear time series analysis*. 2010

360. Campanharo *et al.*, *Duality between Time Series and Networks*. 2011

Given a time series $\mathbf{u} = (u_1, \dots, u_L)^\top$, it is first discretized into $N_\measuredangle \in \mathbb{N}$ *quantile bins*. Each entry u_l, $l = 1, \dots, L$, is assigned to the corresponding bin $\measuredangle_i$, $i = 1, \dots, N_\measuredangle$. A *weighted adjacency matrix* $\mathscr{Z} \in \mathbb{R}^{N_\measuredangle \times N_\measuredangle}$ is then built with entries $z_{i,j} = \overline{z}_{i,j} / \sum_j \overline{z}_{i,j}$, where $j = 1, \dots, N_\measuredangle$ and $\overline{z}_{i,j} \in \mathbb{N}$ is the number of transitions $\measuredangle_i \to \measuredangle_j$ between consecutive time steps — $\mathscr{Z}$ is a *Markov transition matrix*. From a network perspective, each bin represents a node, and each pair of nodes is connected with a weight proportional to the probability that a data point in bin $\measuredangle_i$ is followed by a data point in bin $\measuredangle_j$.

The *MTF encoding* [160] extends $\mathscr{Z}$ by measuring the probabilities of observing a change of value between any pair of points in the time series. The MTF representation also encodes the Markovian dynamics, but the transition probabilities are represented sequentially to avoid losing the time dependence of the conditional relationship. The *MTF matrix* $\overline{\mathscr{Z}} \in \mathbb{R}^{L \times L}$ reads:

$$\overline{\mathscr{Z}} = \begin{bmatrix} z_{i,j} | u_1 \in \measuredangle_i, u_1 \in \measuredangle_j & \dots & z_{i,j} | u_1 \in \measuredangle_i, u_L \in \measuredangle_j \\ z_{i,j} | u_2 \in \measuredangle_i, u_1 \in \measuredangle_j & \dots & z_{i,j} | u_2 \in \measuredangle_i, u_L \in \measuredangle_j \\ \vdots & \ddots & \vdots \\ z_{i,j} | u_L \in \measuredangle_i, u_1 \in \measuredangle_j & \dots & z_{i,j} | u_L \in \measuredangle_i, u_L \in \measuredangle_j \end{bmatrix}, \tag{C.1}$$

which measures the probability of a transition $\measuredangle_i \to \measuredangle_j$ for each pair of time steps, not necessarily consecutive. This is equivalent to spreading out matrix $\mathscr{Z}$ along the time axis by considering the temporal positions of data points in $\mathbf{u}$. By measuring the *quantiles transition probability* at two arbitrary time steps, $\overline{\mathscr{Z}}$ encodes the *multi-span transition probabilities* of the time series.

The MTF requires discretizing the time series into $N_\measuredangle$ quantile bins. Since this discretization is a surjective transformation, it is not reversible and involves the loss of some information. The amount of retained information is mainly controlled by the

refinement level of the discretization. Using an equally spaced discretization, a large $N_{\mathcal{X}}$ might result in a sparse image, which may not effectively highlight structures and patterns in the data, while a small $N_{\mathcal{X}}$ might lead to significant information loss. To achieve a good *trade-off between image sparsity and information loss*, the *symbolic aggregate approximation* algorithm [361] is exploited to perform a *non-uniform bin assignment*. As proposed in [308], the time series is discretized into bins that approximately follow a Gaussian distribution. This non-uniform bin assignment is suitable for handling also time histories that exhibit long-tailed distributions, making the choice of the number of bins less critical. For the experiments presented in Sec. 7.2, the number of bins has been set to $N_{\mathcal{X}} = 20$, which yields satisfactory results without imposing a significant computational burden. Finally, to manage the image size and enhance the computational efficiency of downstream image processing, the MTF matrix $\overline{\mathcal{F}}$ has been downsized by averaging the pixels in each non-overlapping square patch using a *blurring kernel*.

361. Lin *et al.*, *Experiencing SAX: a novel symbolic representation of time series*. 2007

308. Garcia *et al.*, *Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms*. 2022

D IMPLEMENTATION DETAILS OF $\mathrm{NN}_{\mathrm{CAE}}$ AND $\mathrm{NN}_{\mathrm{SUR}}$

In this Appendix, we discuss the implementation details of the DL models employed in Chapter 7. The architectures and training options have been chosen through a preliminary study aimed at minimizing $\mathcal{L}_{\mathrm{CAE}}$ and $\mathcal{L}_{\mathrm{SUR}}$ while retaining the generalization capabilities of $\mathrm{NN}_{\mathrm{CAE}}$ and $\mathrm{NN}_{\mathrm{SUR}}$.

$\mathrm{NN}_{\mathrm{CAE}}$ is set as a 16-layers convolutional autoencoder. The architectures of its encoder $\mathrm{NN}_{\mathrm{ENC}}$ and decoder $\mathrm{NN}_{\mathrm{DEC}}$ are outlined in Tab. D.1a. $\mathrm{NN}_{\mathrm{ENC}}$ is implemented through the composition of four 2D convolutional blocks, each provided with a Softsign-activated *2D convolutional layer* and *2D max pooling*. The number of filter kernels in the 2D convolutional layers is set to $N_F = \{4, 8, 16, 32\}$, with a fixed kernel size of $H_C = 3 \times 3$ across all layers. The output is then reshaped through a *flatten layer* and passed through a Softsign-activated *fully-connected layer* featuring $N_h = 20$ neurons, providing the low-dimensional representation $\mathbf{h}$. This bottleneck layer is symmetrically connected to the decoding branch via a Softsign-activated *fully-connected layer*. The output is reshaped and then processed through four *2D transposed convolutional layers* to reconstruct the input mosaic. The number of filter kernels in the 2D transposed convolutional layers is set to $N_F = \{32, 16, 8, 4\}$, with a fixed kernel size of $H_C = 3 \times 3$. The first three 2D transposed convolutional layers are Softsign-activated, while the output layer uses Sigmoid activation.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}, N_C^{\mathcal{J}}, 1)$	None	–
1 - Conv2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}, N_C^{\mathcal{J}}, 4)$	Softsign	0
2 - MaxPool2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/2, N_C^{\mathcal{J}}/2, 4)$	None	1
3 - Conv2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/2, N_C^{\mathcal{J}}/2, 8)$	Softsign	2
4 - MaxPool2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/4, N_C^{\mathcal{J}}/4, 8)$	None	3
5 - Conv2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/4, N_C^{\mathcal{J}}/4, 16)$	Softsign	4
6 - MaxPool2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/8, N_C^{\mathcal{J}}/8, 16)$	None	5
7 - Conv2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/8, N_C^{\mathcal{J}}/8, 32)$	Softsign	6
8 - MaxPool2D	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/16, N_C^{\mathcal{J}}/16, 32)$	None	7
9 - Flatten	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}} N_C^{\mathcal{J}}/8)$	None	8
10 - Dense	(B_{CAE}, N_h)	Softsign	9
11 - Dense	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}} N_C^{\mathcal{J}}/8)$	Softsign	10
12 - Reshape	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/16, N_C^{\mathcal{J}}/16, 32)$	None	11
13 - Conv2D$^\top$	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/8, N_C^{\mathcal{J}}/8, 16)$	Softsign	12
14 - Conv2D$^\top$	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/4, N_C^{\mathcal{J}}/4, 8)$	Softsign	13
15 - Conv2D$^\top$	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}/2, N_C^{\mathcal{J}}/2, 4)$	Softsign	14
16 - Conv2D$^\top$	$(B_{\mathrm{CAE}}, N_R^{\mathcal{J}}, N_C^{\mathcal{J}}, 1)$	Sigmoid	15

(b)

Filter kernel size:	$H_C = 3 \times 3$
Pool size:	$H_{\mathcal{P}} = \{2, 2, 2, 2\}$
L^2 regularization rate:	$\lambda_{L2} = 10^{-4}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathrm{CAE}} = 128$
Initial learning rate:	$\eta_{\Omega_{\mathrm{CAE}}} = 10^{-3}$
Allowed epochs:	100
Learning schedule:	0.8 cosine decay
Weight decay:	0.05
Early-stop patience:	15 epochs
Train-Val split:	80 : 20
Positive pairings:	$N_+ = 2$
Negative pairings:	$N_- = 2$
Negative margin:	$\overline{\epsilon_{\mathbf{h}}} = 1$

Table D.1: $\mathrm{NN}_{\mathrm{CAE}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

Using *Xavier's weight initialization* [253], NN_{CAE} is trained by minimizing the following *regularized loss function*:

253. Glorot *et al.*, *Understanding the difficulty of training deep feedforward neural networks*. 2010

$$\mathcal{L}^{R2}_{\text{CAE}}(\boldsymbol{\Omega}_{\text{CAE}}, \mathcal{D}^{\text{P}}_{\mathcal{J}}) = \mathcal{L}_{\text{CAE}}(\boldsymbol{\Omega}_{\text{CAE}}, \mathcal{D}^{\text{P}}_{\mathcal{J}}) + \lambda_{L^2} \|\texttt{vec}[\boldsymbol{\Omega}_{\text{CAE}}]\|_2^2, \quad \text{(D.1)}$$

where $\lambda_{L^2} \in \mathbb{R}$ denotes the L^2 regularization rate over the model parameters $\boldsymbol{\Omega}_{\text{CAE}}$. $\mathcal{L}^{R2}_{\text{CAE}}$ is minimized using the first-order stochastic gradient descent optimizer *Adam* [255], for a maximum of 100 allowed epochs.

255. Kingma *et al.*, *Adam: A Method for Stochastic Optimization*. 2015

The *learning rate* $\eta_{\boldsymbol{\Omega}_{\text{CAE}}}$ is initially set to 10^{-3}, and decreased for 4/5 of the allowed training steps using a *cosine decay schedule* with weight decay equal to 0.05. The optimization is carried out considering an 80 : 20 *splitting ratio* of the dataset for training and validation purposes. An *early-stop strategy* is employed to interrupt learning whenever the loss function value on the validation set does not decrease for a prescribed number of patience epochs in a row. The hyperparameters and training options for NN_{CAE} are reported in Tab. D.1b.

NN_{SUR} is a 4-layers *fully-connected DL model*, whose architecture is outlined in Tab. D.2a. The layers feature $\{10, 10, 40, N_h\}$ neurons, respectively. Softsign activations are applied to the hidden layers and no activation is applied to the output layer.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\text{SUR}}, N_\theta)$	None	–
1 - Dense	$(B_{\text{SUR}}, 10)$	Softsign	0
2 - Dense	$(B_{\text{SUR}}, 10)$	Softsign	1
3 - Dense	$(B_{\text{SUR}}, 40)$	Softsign	2
4 - Dense	(B_{SUR}, N_h)	None	3

(b)

L^2 regularization rate:	$\lambda_{L^2} = 10^{-4}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\text{SUR}} = 128$
Initial learning rate:	$\eta_{\boldsymbol{\Omega}_{\text{SUR}}} = 10^{-3}$
Allowed epochs:	5000
Learning schedule:	0.8 cosine decay
Weight decay:	0.01
Early-stop patience:	100 epochs
Train-Val split:	80 : 20

Table D.2: NN_{SUR}: (a) employed architecture, and (b) selected hyperparameters and training options.

The optimization of NN_{SUR} is carried out using *Xavier's weight initialization* together with *Adam* optimizer to minimize the following *regularized loss function*:

$$\mathcal{L}^{R2}_{\text{SUR}}(\boldsymbol{\Omega}_{\text{SUR}}, \mathcal{D}^{\mathbf{h}}_{\boldsymbol{\theta}}) = \mathcal{L}_{\text{SUR}}(\boldsymbol{\Omega}_{\text{SUR}}, \mathcal{D}^{\mathbf{h}}_{\boldsymbol{\theta}}) + \lambda_{L^2} \|\texttt{vec}[\boldsymbol{\Omega}_{\text{SUR}}]\|_2^2. \quad \text{(D.2)}$$

Similarly to what specified for NN_{CAE}, the *learning rate* $\eta_{\boldsymbol{\Omega}_{\text{SUR}}}$ is decreased as the training advances using a *cosine decay schedule*. An *early-stop strategy* is employed to prevent overfitting, with an 80 : 20 *splitting ratio* for training and validation purposes. The relevant hyperparameters and the training options are summarized in Tab. D.2b.

E IMPLEMENTATION DETAILS OF $\mathrm{NN}_{\mathrm{CL}}$ AND $\mathrm{NN}_{\mathrm{RG}}$

In this Appendix, we discuss the implementation details of the DL models employed in Chapter 8. The architectures, as well as the relevant hyperparameters and training options, have been chosen through a preliminary study aimed at minimizing $\mathcal{L}_{\mathrm{CL}}$ and $\mathcal{L}_{\mathrm{RG}}$ while retaining the generalization capabilities of $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}^m_{\mathrm{RG}}$, with $m = 1, \ldots, N_\ell$. Since all $\mathrm{NN}^m_{\mathrm{RG}}$ models share the same architecture, the index m will be dropped in the following for ease of notation.

$\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ are set as 12-layers DL models, whose architecture are outlined in Tab. E.1a and Tab. E.2a, respectively. Both architectures are similar to $\mathrm{NN}_{\mathbf{h}}$, featuring a *damage-sensitive feature extractor* required to be insensitive to transformations in the input not related to damage. This is implemented through the composition of three 1D convolutional blocks, each provided with a Tanh-activated *1D convolutional layer*, *1D max pooling*, and *Dpout regularization*. The number of filter kernels in the 1D convolutional layers is set to $N_F = \{32, 64, 32\}$, in accordance with a proportional scheme $\{N_F, 2N_F, N_F\}$, as suggested in [275], while the corresponding kernel sizes are set to $H_C = \{25, 13, 7\}$, following a scheme $H_C^\ell = H_C^{\ell-1}/2+1$ [358]. The extracted features are then reshaped through a *flatten layer* and run through a stack of three *fully-connected layers* featuring $\{64, 16, N_\ell + 1\}$ and $\{64, 16, 1\}$ neurons, respectively for $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$: the first two are Tanh-activated, while the output layer of $\mathrm{NN}_{\mathrm{CL}}$ is Softmax-activated and the output layer of $\mathrm{NN}_{\mathrm{RG}}$ has no activation function.

275. Wang *et al.*, *Time series classification from scratch with deep neural networks: A strong baseline*. 2017

358. Karim *et al.*, *Multivariate LSTM-FCNs for time series classification*. 2019

Using *Xavier's weight initialization* [253], $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ are trained by minimizing the following *regularized loss functions*, respectively:

253. Glorot *et al.*, *Understanding the difficulty of training deep feedforward neural networks*. 2010

$$\mathcal{L}^{R2}_{\mathrm{CL}}(\boldsymbol{\Omega}_{\mathrm{CL}}, \mathcal{D}_{\mathrm{CL}}) = \mathcal{L}_{\mathrm{CL}}(\boldsymbol{\Omega}_{\mathrm{CL}}, \mathcal{D}_{\mathrm{CL}}) + \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{CL}}]\|_2^2, \quad \text{(E.1)}$$

$$\mathcal{L}^{R2}_{\mathrm{RG}}(\boldsymbol{\Omega}_{\mathrm{RG}}, \mathcal{D}_{\mathrm{RG}}) = \mathcal{L}_{\mathrm{RG}}(\boldsymbol{\Omega}_{\mathrm{RG}}, \mathcal{D}_{\mathrm{RG}}) + \lambda_{L^2} \|\mathtt{vec}[\boldsymbol{\Omega}_{\mathrm{RG}}]\|_2^2, \quad \text{(E.2)}$$

where $\lambda_{L^2} \in \mathbb{R}$ denotes the L^2 regularization rate over the relative model parameters $\boldsymbol{\Omega}_{\mathrm{CL}}$ and $\boldsymbol{\Omega}_{\mathrm{RG}}$. The loss functions $\mathcal{L}^{R2}_{\mathrm{CL}}$ and $\mathcal{L}^{R2}_{\mathrm{RG}}$ are minimized using *Adam* [255], for a maximum of 250 allowed epochs.

255. Kingma *et al.*, *Adam: A Method for Stochastic Optimization*. 2015

The *learning rates* $\eta_{\boldsymbol{\Omega}_{\mathrm{CL}}}$ and $\eta_{\boldsymbol{\Omega}_{\mathrm{RG}}}$ are initially set to $\{10^{-3}, 10^{-4}\}$, and decreased for 4/5 of the allowed training steps using a *cosine decay schedule* with weight decay equal to 0.05. The optimization is carried out considering an 80 : 20 *splitting ratio* of the datasets for training and validation purposes. We use an *early-stop strategy* to interrupt learning, whenever the loss function value attained on the validation set does not decrease for a prescribed

number of patience epochs in a row. The hyperparameters and training options for $\mathrm{NN}_{\mathrm{CL}}$ and $\mathrm{NN}_{\mathrm{RG}}$ are reported Tab. E.1b and in Tab. E.2b, respectively.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{CL}}, L, N_u)$	None	–
1 - Conv1D	$(B_{\mathrm{CL}}, L, 32)$	Tanh	0
2 - MaxPool1D	$(B_{\mathrm{CL}}, L/2, 32)$	None	1
3 - Dropout	$(B_{\mathrm{CL}}, L/2, 32)$	None	2
4 - Conv1D	$(B_{\mathrm{CL}}, L/2, 64)$	Tanh	3
5 - MaxPool1D	$(B_{\mathrm{CL}}, L/4, 64)$	None	4
6 - Dropout	$(B_{\mathrm{CL}}, L/4, 64)$	None	5
7 - Conv1D	$(B_{\mathrm{CL}}, L/4, 32)$	Tanh	6
8 - MaxPool1D	$(B_{\mathrm{CL}}, L/8, 32)$	None	7
9 - Dropout	$(B_{\mathrm{CL}}, L/8, 32)$	None	8
10 - Flatten	$(B_{\mathrm{CL}}, 4L)$	None	9
11 - Dense	$(B_{\mathrm{CL}}, 64)$	Tanh	10
12 - Dense	$(B_{\mathrm{CL}}, 16)$	Tanh	11
13 - Dense	$(B_{\mathrm{CL}}, N_{\mathscr{E}} + 1)$	Softmax	12

(b)

Filter kernel size:	$H_C = \{25, 13, 7\}$
Pool size:	$H_{\mathcal{P}} = \{2, 2, 2\}$
Dropout rate:	5%
L^2 regularization rate:	$\lambda_{L^2} = 10^{-3}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathrm{CL}} = 32$
Initial learning rate:	$\eta_{\Omega_{\mathrm{CL}}} = \{10^{-3}, 10^{-4}\}$
Allowed epochs:	250
Learning schedule:	0.8 cosine decay
Weight decay:	0.05
Early-stop patience:	15 epochs
Train-Val split:	80 : 20

Table E.1: $\mathrm{NN}_{\mathrm{CL}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

(a)

Layer	Output shape	Activation	Input
0 - Input	$(B_{\mathrm{RG}}, L, N_u)$	None	–
1 - Conv1D	$(B_{\mathrm{RG}}, L, 32)$	Tanh	0
2 - MaxPool1D	$(B_{\mathrm{RG}}, L/2, 32)$	None	1
3 - Dropout	$(B_{\mathrm{RG}}, L/2, 32)$	None	2
4 - Conv1D	$(B_{\mathrm{RG}}, L/2, 64)$	Tanh	3
5 - MaxPool1D	$(B_{\mathrm{RG}}, L/4, 64)$	None	4
6 - Dropout	$(B_{\mathrm{RG}}, L/4, 64)$	None	5
7 - Conv1D	$(B_{\mathrm{RG}}, L/4, 32)$	Tanh	6
8 - MaxPool1D	$(B_{\mathrm{RG}}, L/8, 32)$	None	7
9 - Dropout	$(B_{\mathrm{RG}}, L/8, 32)$	None	8
10 - Flatten	$(B_{\mathrm{RG}}, 4L)$	None	9
11 - Dense	$(B_{\mathrm{RG}}, 64)$	Tanh	10
12 - Dense	$(B_{\mathrm{RG}}, 16)$	Tanh	11
13 - Dense	$(B_{\mathrm{RG}}, 1)$	None	12

(b)

Filter kernel size:	$H_C = \{25, 13, 7\}$
Pool size:	$H_{\mathcal{P}} = \{2, 2, 2\}$
Dropout rate:	10%
L^2 regularization rate:	$\lambda_{L^2} = 10^{-3}$
Weight initializer:	Xavier
Optimizer:	Adam
Batch size:	$B_{\mathrm{RG}} = 32$
Initial learning rate:	$\eta_{\Omega_{\mathrm{RG}}} = \{10^{-3}, 10^{-4}\}$
Allowed epochs:	250
Learning schedule:	0.8 cosine decay
Weight decay:	0.05
Early-stop patience:	15 epochs
Train-Val split:	80 : 20

Table E.2: $\mathrm{NN}_{\mathrm{RG}}$: (a) employed architecture, and (b) selected hyperparameters and training options.

COLOPHON

This Thesis has been typeset using LaTeX and complied with `pdflatex`.
The Thesis template is due to M. Opgenoord /phthesis-template.

Several books have influenced the style, typography, and graphic design of this thesis, in particular:

Edward Tufte's *Beautiful Evidence* (2006)

Jean-Luc Doumont's *Trees, Maps, and Theorems* (2009)

Robert Bringhurst's *Elements of Typography* (1992)

Model-Based and Data-Driven Methodologies
Toward Predictive Digital Twins of Structures

Matteo Torzoni, May 2024

© Politecnico di Milano. All rights reserved.

Doctoral Dissertation
Matteo Torzoni

Model-Based and Data-Driven Methodologies
Toward Predictive Digital Twins of Structures

– MMXXIV –