POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Uncertainty propagation in experimental data pipelines

Master of Science in Computer Science and Engineering

Author: **Pasquale Dazzeo**

Student ID: 965117
Advisor: Prof. Barbara Pernici
Co-advisor: Ing. Edoardo Ramalli
Academic Year: 2022-23

Abstract

Data analysis has become a crucial process in various scientific fields, as the increasing availability of technological tools facilitates decision-making through data-driven models. The complex task of processing data leading to final decisions is often implemented in multi-stage pipelines, each assigned to handle different stages of data manipulation. In scientific contexts, it is common to build models to abstract physical phenomena using experimental data from different experiments. However, experimental data are often uncertain and irreproducible, and their processing within pipelines is necessary for the construction of reliable models. As a result, data preparation techniques have been employed and refined over the years to improve Data Quality, which is critical to obtaining a good model. This thesis proposes an approach aimed at the artificial generation of uncertainty in data through Fault Injection, a common method in the field of Data Quality. Uncertain data is then given as input to a multi-stage pipeline, and the ambiguity generated in the output is quantified. Uncertainty propagation is evaluated, showing the relationship between uncertainty-generating factors and ambiguity in the output. The case study of this work focuses on the complex multi-stage pipeline of the Curve Matching (CM) framework, which measures the similarity between two curves and assesses the agreement between experimental data and corresponding simulation obtained from model prediction. CM is characterized by ambiguity, which is due to the randomness of some processes, uncertainty, and Data Quality issues. Thus, this thesis aims to help the pipeline user understand the sources generating ambiguities and their impact on the pipeline and each stage. The proposed approach quantifies the impact of Fault Injection on both individual stages and the entire pipeline, providing a comprehensive analysis of the impact and propagation of uncertainty generated by various factors. The analysis tool developed through this thesis will help the pipeline user understand the robustness of the results against different uncertainty and Data Quality conditions in the experimental data.

Keywords: multi-stage pipeline; model validation; curve matching, uncertainty propagation, data quality

Sommario

L'analisi dei dati è diventata un processo cruciale in vari campi scientifici, poiché la crescente disponibilità di strumenti tecnologici facilita il processo decisionale attraverso modelli basati sui dati. La complessa attività di elaborazione dei dati che porta alle decisioni finali è spesso implementata in pipeline a più stadi, ognuno dei quali è incaricato di gestire diverse fasi di manipolazione dei dati. Nei contesti scientifici, è comune costruire modelli per astrarre i fenomeni fisici utilizzando dati sperimentali provenienti da esperimenti diversi. Tuttavia, i dati sperimentali sono spesso incerti e irriproducibili e la loro elaborazione all'interno di pipeline è necessaria per la costruzione di modelli affidabili. Di conseguenza, tecniche di preparazione dei dati sono state impiegate e perfezionate nel corso degli anni per migliorare la qualità dei dati, che è fondamentale per ottenere un buon modello. Questa tesi propone un approccio finalizzato alla generazione artificiale di incertezza nei dati attraverso la Fault Injection, un metodo comune nel campo della Data Quality. I dati incerti vengono quindi dati in ingresso a una pipeline a più stadi e l'ambiguità generata nell'output viene quantificata. Viene valutata la propagazione dell'incertezza, mostrando la relazione tra i fattori che generano incertezza e l'ambiguità in uscita. Il caso di studio di questo lavoro si concentra sulla complessa pipeline multi-stadio del framework Curve Matching (CM), che misura la somiglianza tra due curve e valuta l'accordo tra i dati sperimentali e la simulazione corrispondente ottenuta dalla previsione del modello. Il CM è caratterizzato da ambiguità, dovuta alla casualità di alcuni processi, all'incertezza e a problemi di qualità dei dati. Pertanto, questa tesi si propone di aiutare l'utente della pipeline a comprendere le fonti che generano ambiguità e il loro impatto sulla pipeline e su ciascuna fase. L'approccio proposto quantifica l'impatto della Fault Injection sia sulle singole fasi che sull'intera pipeline, fornendo un'analisi completa dell'impatto e della propagazione dell'incertezza generata da vari fattori. Lo strumento di analisi sviluppato in questa tesi aiuterà l'utente della pipeline a comprendere la robustezza dei risultati rispetto a diverse condizioni di incertezza e qualità dei dati sperimentali.

Parole chiave: pipeline multi-stadio; validazione modelli; curve matching; propagazione dell'incertezza; qualità dati

Contents

1 | Introduction

In the digital age, data has become a critical resource for scientific research and business operations. However, the abundance of available data does not always translate into quality and reliability. Data often needs to be pre-processed and cleansed to remove erroneous and unusable information: thanks to these processes, it is possible to build models of the highest possible quality. Having good data is therefore critical to have good models, but it is not enough to guarantee high performance. Models need to be validated with experimental data to ensure their reliability and usability in real-world scenarios.

Experimental data is uncertain and unreliable due to various factors such as measurement errors. Data pipelines have emerged as a means to address these issues by providing a streamlined and automated approach to data and model management. Various types of processes aimed at preprocessing and manipulating data for model validation take place in these pipelines. However, the operations performed in the pipeline may introduce errors and ambiguity, and the user is often unaware of these problems.

The approach developed in this thesis is based on the simulation of faults in the data to evaluate how each process in the pipeline responds to them and to assess the impact on the final output of the pipeline. By injecting errors into the data at different stages of the pipeline, it is possible to see where the errors are introduced, how they propagate, and how they affect the quality of the final output.

Pipelines consist of a series of interconnected processes that work together to transform and analyze data, with the ultimate goal of producing accurate, high-quality models.

Despite their usefulness, it is widely recognized that errors can be introduced into pipeline processes and affect the outcome.

By introducing uncertainty into the data through several factors at different stages of the pipeline, users can identify which stages are most susceptible to errors and understand how these errors propagate through the pipeline.

The goal of this approach is to help pipeline users understand the potential sources of error within the pipeline and to help them optimize the pipeline to improve the quality of the resulting models. In particular, the obtained can be used to evaluate the impact of uncertainties in experimental data.

Having high-quality data is necessary to have high-quality models: however, if it is experimental data, there is not much to be done as they can generally be uncertain or irreproducible for a variety of reasons. This is also reflected in the validation of models, which is carried out with experimental data and is subject to their quality of them [1].

Data pipelines manage automated approaches to carry out data management and model validation processes, used in a variety of fields such as data science or machine learning. However, operations that occur within these fundamentally important pipelines can introduce additional uncertainty that results in the ambiguity of the output.

Identifying these errors and understanding their propagation process can be critical in improving data and model management processes.

The subject of this thesis will be Curve Matching [2], a framework used for validating models against experimental data: it can compare curves generated by experimental data and models, returning an index of similarity between them.

The results that this method produces, however, can sometimes be ambiguous: when dealing with curves that are very similar to each other, it is difficult to distinguish one best model among all or to understand where one model is worse than another.

It is possible to see Curve Matching as a multi-stage pipeline where experimental data are received as input and a similarity score is returned as output. The stages are shown in Figure 1.1 group several operations that are used to manipulate the data.

During the research, it was identified that ambiguity in outputs is created by various factors throughout the operations occurring in the pipeline: to tackle this issue, *fault injection*, a widely used technique in the field of Data Quality [3], was used. This technique allowed to introduce *uncertainty* into the data and analyze how it propagates into the pipeline, resulting in ambiguous outputs.

The developed approach consists of the computation of metrics to quantify the *ambiguity* generated by the faults injected into the pipeline. This approach allowed an understanding of the relationship between the analyzed metrics and the faults injected, ultimately leading to a better understanding of the factors that contribute to *ambiguity* in the outputs.

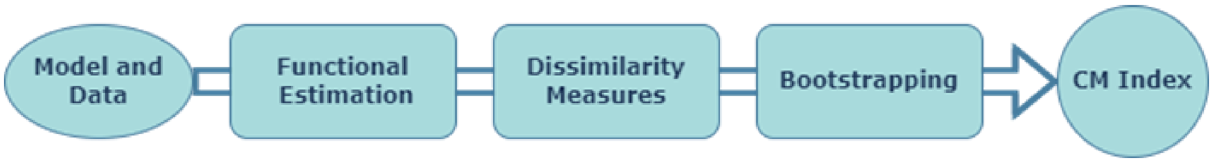


Figure 1.1: Curve Matching pipeline with three stages.

The developed approach is similar to the one applied by the author of [4], that applied fault injection on Curve Matching, using the final score to quantify the ambiguity introduced by uncertain data.

Resuming his work, an application of fault injection extended to individual stages of the Curve Matching pipeline is presented in this thesis. In addition, combined with the measurement of output ambiguity, the relationship between ambiguity and uncertainty factors is quantified.

By introducing perturbations into the data at different stages of the pipeline, users can identify which aspects of the data are most sensitive to error and optimize the pipeline accordingly.

This paper's contribution to research will be the introduction of a tool for analyzing uncertainty within experimental data pipelines using fault injection. Compared to analyses such as [4], several uncertainty-generating variables are introduced. The analysis is also carried out at the level of individual stages through the use of a general metric to quantify ambiguity in the outputs of each stage. Uncertainty propagation shows the pipeline using the processes that are most sensitive to poor-quality data and thus provides useful recommendations for improving the pipeline design.

The structure of the thesis is as follows.

Chapter 2, introduction to key concepts in the state of the art. The concepts of Model Validation and the various existing techniques, modeling uncertainty in data and models, data pipelines and the transformations that can occur in data within them, the concept of Data Quality, and the fault injection technique are introduced.

Chapter 3, overview of the research process and methodology design. The research questions and the context in which the methodology is developed, are highlighted. Also, the design of the methodology including the core elements of fault injection, uncertainty assessment, and propagation are explained.

Chapter 4, contains a description of the implementation of the methodology, explaining the algorithms used to implement the design presented in Chapter 3. Also, the process of data collection from the SciExpeM framework[5] and the technologies used to analyze experimental data are presented.

Chapter 5, contains the results of the application of the methodology developed on the Curve Matching pipeline. The results of the two main steps of Uncertainty Assessment and Propagation are presented, in response to the research questions explained in Section 3.1.

Chapter 6, contains a summary of the results achieved through the application of the methodology, its weaknesses, and insights for future research in the context of the work.

2 | State of The Art

In this chapter, the concepts of Model Validation, Uncertainty Modelling, Data Quality and Data Pipeline will be presented, which are crucial for understanding the general framework on the tools currently available to address various problems in the validation of complex models that utilise multi-stage pipelines to process the available data.

In Section 2.1, the validation metrics, necessary to establish the validity and reliability of predictive models, are presented, as well as the Curve Matching index and the entire process leading to its calculation.

Section 2.2 will delve into the concept of modelling uncertainties, which are inherent in all predictive models and obviously in data itself. The importance of the presence of uncertainty and some techniques used to quantify it, and how it propagates from the data to the final model will be discussed.

Then, in Section 2.3, the concept of Data Quality will be presented, a critical component in the development of predictive models and, as we will see, a powerful tool for analysing the quality of the data at our disposal and the analyses done. The factors that contribute to poor Data Quality will be discussed and the main dimensions included in the proposed analysis will be analysed.

In Section 2.4, an overview of the data pipeline and the processes that take place within it and how they impact the final result will be discussed. In addition, an overview will be given of the possible transformations applicable to the data within the pipeline.

Thus, to understand the key approach used in the methodology, Section 2.5 will present the fault injection, a technique used to test the sensitivity of a system to the presence of errors in the data, commonly used in Data Quality studies. Using this technique, one is able to establish the robustness of a system with respect to varying degrees of data deterioration.This technique is a very useful tool for an analyst because, especially in the case of data that are not readily available, it is able to simulate a variety of inputs that are otherwise not reproducible.

The topics discussed in this chapter will serve to provide an overview of the key areas that form the backbone of the entire validation process of a predictive model and how data are used within it. It is important to bear in mind that the approaches discussed are broad and provide a general way of handling the situation.

This clearly makes it impossible to establish a precise methodology for the analysis and impact of uncertainty on the validation of each predictive model, but it does allow the identification of a set of principles to appeal to, such as the analysis of the various dimensions of Data Quality when subjected to different levels of uncertainty.

2.1. Model Validation

Model validation is an important part in the overall model development procedure: it verifies that, within its domain of applicability, the assumptions made in the idealization and generation of the model yield a satisfactory prediction.

However, the term satisfactory takes on a broader meaning because each model has a specific context and its validity depends on comparisons between his predictions and results obtained by certain experiments. Moreover, without a way to objectively quantify the model performance, different experts can have discordant assessments regarding the model validity. In both cases, with an objective or not objective way to quantify the model validity, the number and diversity of the experiments have also an impact [6].

Before a validation comparison can be executed there are decisions and criteria to be well-defined, such as the use of the model and its purpose, validation experiments, metrics and requirements, domain of comparison and calibration metrics; in literature, there are plenty of useful model validation guidelines that include best practices to the development of a successful validation procedure [5, 7, 8].

After stressing the importance of having a plan in place to validate a model, Paez et al. [7] propose a comprehensive framework with activities to carry out model calibration and validation experiments. In addition, before the experimentation, modeling and validation activities begin, a set of best-practices to be followed and criteria to be defined are defined. After that, model experimentation activities are specified, at the end of which it is decided whether the model is valid and therefore can be used, otherwise possible causes of model invalidity are indicated and the model can be recalibrated. However, the validation procedures carried out through these guidelines leave much to the judgment of analysts, experimentalists, and stakeholders. The validation plan should be created cooperatively by all, as well as the validation requirements should be agreed upon among all parties, and furthermore, the validation activities should be performed by independent groups and then compared with each other.

In this regard, the Guide for Verification and Validation in Computational Solid Mechanics, of which Schwer [8] offers an overview. The key takeaway from this work is that the guide represents a foundation document for the Verification & Validation (V&V) of computational models, since in such a subject achieving a step-by-step standard is difficult, V&V being a science where different points of view have value and should not be evaluated as right or wrong. Therefore, like [7], guidelines directed to analysts, experimentalists, code developers and physics model developers are outlined in this document. The key principles of the guidance are:

- Verification must precede validation
- The need for validation experiments and accuracy requirements for computational model predictions depend on the use of the model and must be part of V&V activities
- Validation of a complex system should be performed in a hierarchical fashion
- Validation is specific to a particular computational model for a particular intended use
- Validation must assess the predictive ability of the model in the physical realm of interest and be able to account for uncertainties that arise from simulation results and experimental data

In both Paez and Schwer, the need for a model verification process, which takes place before validation, is emphasized. The verification phase is used to determine that a computational model accurately represents the underlying mathematical model and its solution. Verification occurs in basically two parts:

- Code verification, to determine that the mathematical model and the algorithm that implements it work correctly.
- Verification of the calculations to establish that the discrete solution of the mathematical model is correct.

In short, the verification part covers the mathematical domain while the validation part covers the physical domain. Finally, the importance of the role of Uncertainty Quantification (UQ) for both modelers and experimentalists is stressed in both guidelines. For it is common to perform more than one experiment and produce somewhat different results, just as each computation includes numerical and physical parameters that have different distributions of values: the role of UQ is to quantify the variability of experimental results and the effect of simulation parameters on the final result.

It goes without saying that an improper validation leads to various problems like poor performance on unseen data, which is the ultimate purpose of the model itself, or unreliable outputs that perform well on the samples on which the model is built but fail in different scenarios; thankfully, in the last two decades the availability of experimental data and the development of data sharing systems allowed the development of very accurate and sophisticated predictive models in a variety of scientific fields where often the impossibility of acquiring experimental data has been a barrier in modeling certain systems.

2.1.1. Qualitative techniques

Visualization

The introduction of graphical representations for computational models has meant that one of the best validation techniques is precisely the visualization of their graphical representation: this is because the amount of time and resources required to carry out such an evaluation is minimised.

Basically, for a team of experts, establishing the goodness of the model based on its visualization is a relatively quick and easy task.

However, with the advent of ML and DL together with big data, a number of issues, such as the quantification of uncertainty and its impact on increasingly complex models, as well as the breadth of scientific applications of the models, required the introduction of metrics that quantitatively assert the actual validity of the model.

Furthermore, since a qualitative validation relies on the opinion of experts, it is not certain that the latter will always agree on the actual assessment of the model's goodness.

2.1.2. Quantitative techniques

Point-wise approach

Point-wise approaches arise as a natural solution to the problems of ambiguity in visualisation techniques: they are able to return an accurate measurement of the agreement between the model and empirical observations by defining score functions that measure the error between the prediction and the experimental data point by point [5].

Despite the speed of calculation and ease of use in many applications, these approaches lack an important feature: in an experimental context, the points belonging to a data set are a sequence of measurements of a specific physical phenomenon, so they are expected to follow some kind of trend.

Therefore, a measurement of the validity of the model based simply on the distance between the points cannot express how well the model was able to emulate the real-world system for which it was constructed [2, 5].

As stressed earlier, data collected from experiments, then used to build predictive models, have inherent uncertainty and show systematic errors. Running the models shows the presence of misleading data (so-called outliers) and allows experiments to focus on certain problem areas, where modelers need to be more careful in calibrating the parameters of a model.

On the other hand, any errors present in the model's predictions could result from a deficiency of the modelers in setting the parameters and not from a deficiency of interpretation by the model itself.

Score functions weighted on the experimental uncertainties, namely Error Function Value (EFV) and absolute deviation (D), are used to assess the goodness of models with respect to experimental data:

$$EFV = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{n_i}\sum_{j=1}^{n_i}\left(\frac{Y_{ij}^{sim} - Y_{ij}^{exp}}{\sigma\left(Y_{ij}^{exp}\right)}\right)^2\right] \tag{2.1}$$

$$D = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{n_i}\sum_{j=1}^{n_i}\left(\frac{Y_{ij}^{sim} - Y_{ij}^{exp}}{\sigma\left(Y_{ij}^{exp}\right)}\right)\right] \tag{2.2}$$

being

$$Y_{ij} = \begin{cases} Y_{ij} & \text{if } \sigma(Y_{ij}^{exp}) \cong constant \\ ln(Y_{ij}) & \text{if } \sigma(ln(Y_{ij}^{exp})) \cong constant \end{cases} \tag{2.3}$$

where N and n_i are respectively the number of datasets and number of data within the i-th dataset. Y_{ij}^{exp} is the j-th experimental observation within each i-th dataset and $\sigma(Y_{ij}^{exp})$ its standard deviation. A full explanation can be found in the Supplementary Material of the corresponding work [9].

After collecting the experimental and simulation data, for each type of experiment considered (2.1) and (2.2) were computed in order to calculate an overall score for each type of model and find the best one [9, 10].

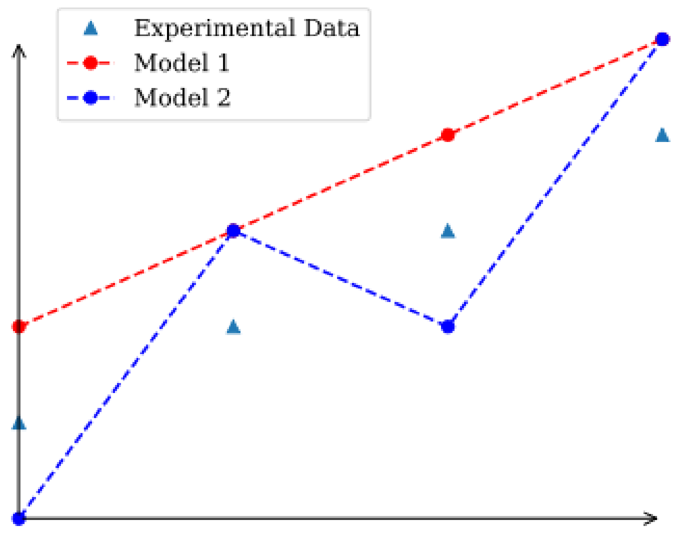


Figure 2.1: Drawback of point-wise approaches: indeed, such a score function would return the same score for both models despite the fact the shapes are significantly different[11].

This paper shows how comparing three models qualitatively (i.e., graphically observing which one performs better) and quantitatively yields outcomes that are quite the opposite.

Not only does a validation approach based on two scoring functions that also account for experimental uncertainties fail to identify the best model, but it also fails to provide useful information for modelers to calibrate the models.

Beware however, as has already been mentioned, each type of validation process concerns a specific model: for example in the context of ML models, such as linear regression, score functions are an excellent way of performing model validation and selection.

Among these, the most famous is the Sum Squared Error (SSE):

$$SSE = \sum_{n=1}^{n} (y_i - f(x_i))^2 \tag{2.4}$$

where y_i is the i-th empirical observation, x_i the i-th explanatory variable and $f(x_i)$ the predicted value of y_i.

The previously introduced scoring functions, EFV and D, were based on the SSE (2.4), but also account for the experimental uncertainty inherent in the measurements and nevertheless fail to provide any useful insight within the context of their application.

This does not mean that these functions are useless, but rather emphasises the fact that the reliability of a validation metric, no matter how robust and easy to apply it may seem, also depends on other factors: for example in the case of EFV (2.1) and D (2.2), although they also take into account the experimental uncertainty indicated as the standard deviation of the data, the most obvious limitation is due to the fact that they do not take into account the trend followed by the experiments and base their evaluation solely on distance-based point-to-point measurements.

Thus, the use of score functions as a validation metric is justified by their speed of computation but misses in capturing a fundamental aspect: the points come from a set of experimental measurements and thus follow a trend coming from the physical phenomenon that the model must abstract.

Figure 2.1 clarifies the concept: even though the trend shown by the points predicted by Model 1 is different from the trend of the points predicted by Model 2, the point-wise error of the models when computed against experimental data is identical.

Mean Absolute Error

The Mean Absolute Error is a commonly used metric for evaluating the performance of machine learning models. It measures the average absolute difference between the predicted values and the actual values.

The metric will be used in the development of the methodology to measure the error of the pipeline output with an ideal input and a corrupted input. In this case it will only be of interest to have a metric for comparing the degree of uncertainty introduced into the pipeline. The formula used for MAE is:

$$MAE = \frac{\sum_i^N |\widehat{y_i} - y_i|}{N}$$

where N will be the number of experimental observations in an experiment, y_i the output value of the pipeline stage under analysis obtained under the original conditions, and $\widehat{y_i}$ the output value obtained under different conditions.

2.1.3. Curve Matching

To account for the trend inherent in experimental measurements of certain phenomena, the authors of [2] presented a framework for comparing models and experiments called Curve Matching.

This approach returns an index in the [0,1] range to evaluate the similarity of the model simulation with respect to experimental observations.

The following subsections will present the Curve Matching methodology implemented in the work of [12], where the authors propose a data ecosystem for the management of scientific experiments, where CM is used as a global measure of model performance.

The methodology proposed in this thesis is developed on this approach, as in the entire framework the data undergoes various transformation processes aimed at generating a measure of similarity between the model and the experimental samples, which represents the degree of validity of the model itself: in particular, the focus of the work is on the amount of uncertainty introduced in the various stages of the pipeline, and its propagation up to the final stage.

The overall goal is to return a quantitative measure of uncertainty on the output, as this will denote the level of reliability of the used validation metric.

Figure 2.2 shows the various stages of the pipeline together with the sources of uncertainty. Note that experimental uncertainty, although not always indicated in the data, is inherent in the measurements, just as aleatoric and epistemic uncertainty is in the models.

Furthermore, the stages that have to do with data manipulation are indicated as possible sources of uncertainty because the transformations that occur within them are subject to certain parameters that, combined with the uncertainties of the initial stages, will affect the accuracy of the final index.

A further clarification must be made: Figure 2.2 shows a 'conceptual' pipeline of Curve Matching to show the functionality of the system at a high level.

The actual implementation in [12] takes place via Python libraries, whose functionalities of interest will be discussed in more detail later.

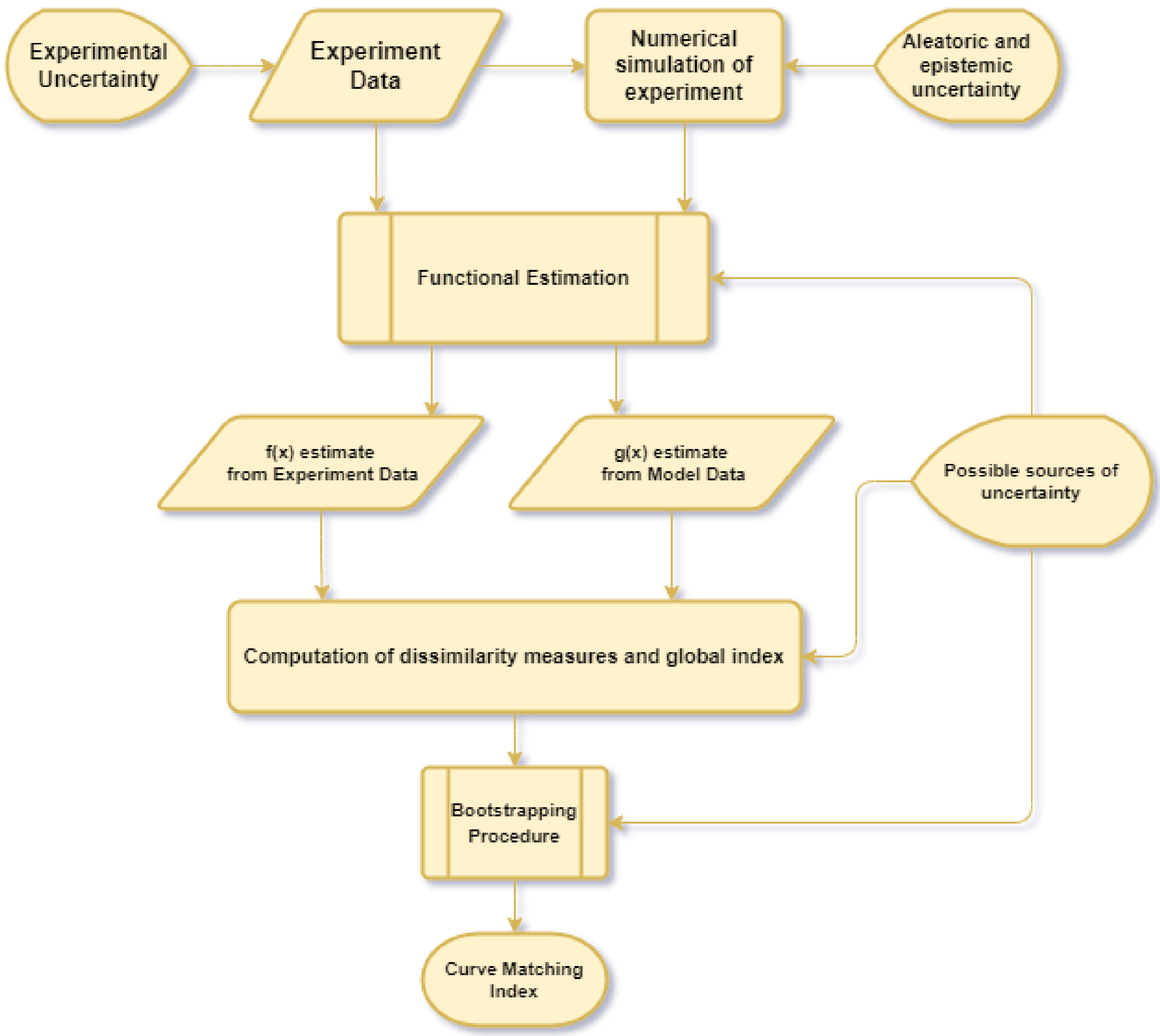

Experimental Uncertainty
Experiment Data
Numerical simulation of experiment
Aleatoric and epistemic uncertainty
Functional Estimation
f(x) estimate from Experiment Data
g(x) estimate from Model Data
Possible sources of uncertainty
Computation of dissimilarity measures and global index
Bootstrapping Procedure
Curve Matching Index

Figure 2.2: An high-level representation of the Curve Matching pipeline in the SciExpeM framework [5].

Functional Estimation

In our application of interest, in order to obtain a similarity score, the experimental data and associated model simulations are represented as square integrable functions and have square integrable first derivative: these conditions are necessary for the calculation of the dissimilarity indices that contribute to the final score.

To obtain the functional estimation, the spline smoothing approach with a roughness penalty presented in the work of Ramsay and Silverman [13] is used. Further explanations on the principles of functional estimation can be found in Ramsay's work [14].

A more in-depth look at the spline interpolation process is necessary, as the functional estimates of the experimental and model data, f(x) and g(x) respectively, are used to represent the process from which the data is generated, and based on this estimate the similarity index will subsequently be calculated: the entire Curve Matching methodology relies on the representational capacity of splines, so it is important to understand how this approach works in order to assess whether the final result can be trusted.

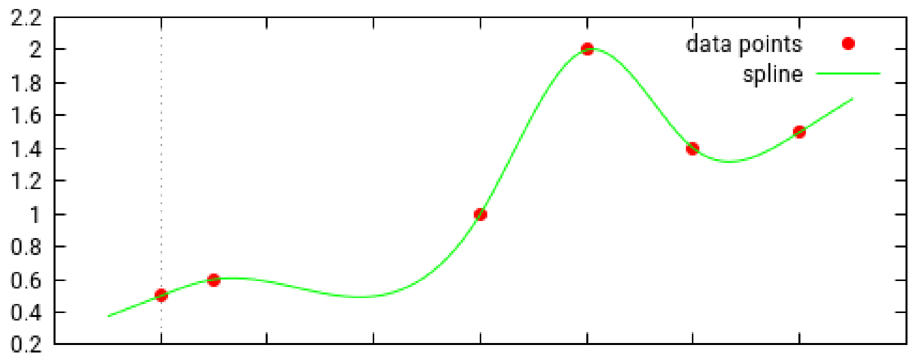


Figure 2.3: An example of cubic spline, taken from the website [15] of an open-source C++ library for spline interpolation.

To understand the implementation of Curve Matching in [12], which will be the subject of this study, it is therefore necessary to introduce the mathematical concepts underlying the methodology, starting with the definition of spline.

In mathematics, a spline is a special function defined piecewise by polynomials and is preferred to classical polynomial interpolation approaches as it leads to similar results with low-degree polynomials, avoiding the problems associated with the use of high-degree polynomials [16]: a piecewise defined function is defined by several sub-functions, each defined in a subinterval of the function's domain.

The points at which the domain is divided into sub-intervals are called knots: the choice of knots is crucial in the application of interest, since the shape of the spline is strongly dependent on their position.

Hence, we need to introduce the spline smoothing methodology presented by Ramsay and Silverman [13] which will later be used as a method to produce functional estimates for the Curve Matching framework.

Spline smoothing method estimates a curve x from experimental observations by minimizing two conflicting goals in curve estimation.

The first goal ensures that the estimated curve gives a good fit to the data, by means of the SSE, in contrast to the second goal aiming to have a fit that is not perfect, as it may result in a too wiggly curve. The latter requirement is formalized in the roughness penalty defined as the integrated squared second derivative of a function.

The use of this approach is due to the fact that obtaining estimates that vary smoothly from one value to the next is equivalent to taking information from every neighbouring point: this concept materialises the need to obtain functional estimates capable of approximating the regularity of the processes from which the data come.

This is the fundamental assumption that is also made in the presentation of the CM framework [2]: data are noisy point-wise evaluations of an underlying smooth functional process, whose realization is estimated from data as a pre-processing step.

Now, for instance, if (x_i, y_i) for i $\epsilon \{1, .., n\}$ are the point-wise experimental data or evaluations of the model, the smooth functional estimate $\hat{f}$ for these data is obtained by minimizing the objective function called Penalized Sum of Squares:

$$\hat{f} = argmin_{f \in F} \left[\sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int (f^{''}(x))^2 dx \right]_{\lambda > 0} \tag{2.5}$$

where F is the space of spline functions with a fixed polynomial degree and a fixed number of knots.

A remarkable theorem found in de Boor (2002) [17] states that the curves minimizing (2.5) are cubic splines with knots at the data points: a more detailed explanation is provided in [26,29].

With regard to the application of Curve Matching, in the original paper [2] the F-space contains splines with a polynomial degree of 5, whereas the implementation of CM in the SciExpeM framework [12], the one analysed in this thesis, the splines, consistently with the aforementioned De Boor theorem, have a polynomial degree of three.

The choice of nodes in splines is very important, as the shape of the spline depends on their position. Various techniques for knots placement exist in the literature: in this case, their position depends on the number of initial points available and the data source (model or experimental data).

Without going into too much detail on the implementation, it is sufficient to know that there are two basic cases: depending on the number of points, the nodes are either positioned at the same location as the data or are chosen to be equidistant along the percentiles of data locations.

Figure 2.4 show two splines calculated from experimental data: as can be seen, the trend of the data is not always correctly captured by the spline.

The lack of a fixed, reliable method feeds the uncertainty in the positioning of the knots and will have a negative effect on the validation of the model: indeed, it is not possible to produce a similarity score between two curves that is reliable if the functional estimate is uncertain.

In (2.5), one can clearly distinguish the two terms of SSE and roughness: λ, the smoothing parameter, adjusts the degree of fit to the data and the variability of the function, represented by the first and second terms respectively. Figure 2.5 shows how the choice of λ affects the spline. The choice of λ is crucial since small values of λ provide under-smoothing of the noisy data, while large values produce over-smoothed curves.

The smoothing parameter is chosen according to a variant of the classic Generalized Cross-Validation (GCV) method for calculating smoothing splines [13]. According to this criterion, the optimum λ is the one that returns the optimum value of the GCV function:

$$\lambda_{opt} = argmin_{\lambda \in \mathbb{R}^+} GCV(\lambda) \tag{2.6}$$

where

$$GCV(\lambda) = \frac{n \sum_{i=1}^{n} (y_i - f(x_i))^2}{(n - df(\lambda))^2} \tag{2.7}$$
$$df(\lambda) = trace(S)$$

being S the smoothing matrix while df(λ) is a measure of the degrees of freedom of the spline [13].

The calculation of the Curve Matching index depends on four indices, which are also calculated from the approximation of the function's first derivative. Since Equation (2.7) often does not return a good first derivative approximation, a modified version of the GCV is used in CM [2]:

$$GCV_1(\lambda) = \frac{n \sum_{i=1}^{n-1} \left(y_i' - \widehat{f}'(x_i) \right)}{(n - df(\lambda))^2} \tag{2.8}$$

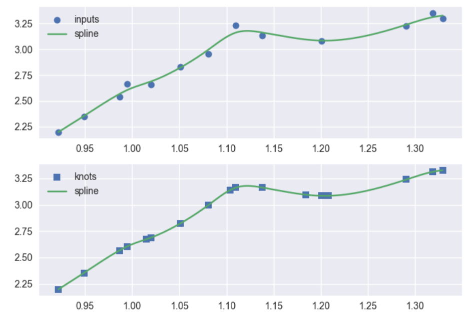
$\widehat{f}'$ is the first derivative of (2.5) and y_i' for i$\in$ $(2, ..., n-1)$ are the estimates of the first derivative of data. In Figure 2.6 it's clear that GCV_1 returns a good approximation of both the function and its first derivative. In the left panels, λ is obtained by minimizing Equation (2.7) whereas in the right panels minimizing Equation (2.8).

The figures in the top row highlight the points and splines, while those in the bottom row show the centred finite differences of the data points and the derivative before the spline. It is clear that a better approximation of the first derivative is obtained with the modified version of GCV.

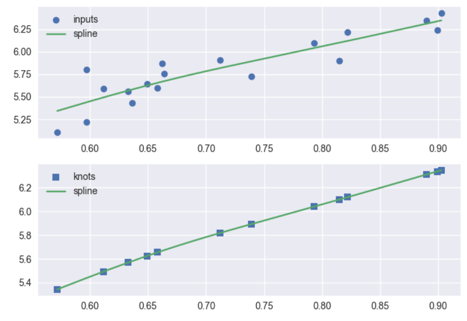
Thus, at the end of the first stage of the pipeline, the data were manipulated to produce a functional estimate representing the underlying process generating them.

Before entering the second stage, it is useful to fix the key concept to be remembered in the perspective of this work. The algorithm that generates the optimal spline solves an optimisation problem, where the main objective is to find a curve that minimises the objective functions Equations (2.5), (2.6) and (2.8): these calculate scores for each curve in the F-space of Equation (2.5) and return the best one.

The step to which the greatest uncertainty is attributed, therefore, is the positioning of the nodes of the splines, on which the population of the F-space depends: we can in fact see from Figure 2.4a that the spline has knots whose trend strongly resembles that of the experimental points, while in Figure 2.4b the optimal spline, which among the possible ones will have the best PENSSE value, is in any case not satisfactory since the selected knots follow a linear trend, quite far from that of the observations.



(a) First spline.



(b) Second spline.

Figure 2.4: Two splines computed with experimental data where data and knot placement is specified.

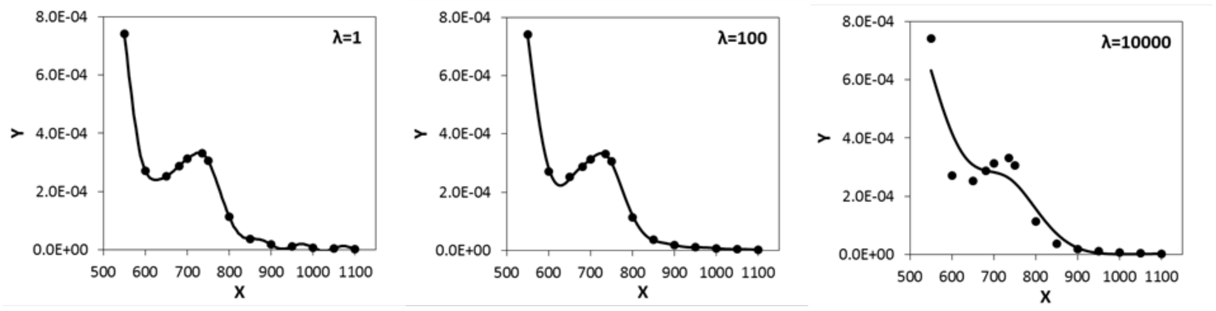


Figure 2.5: Smoothing splines over the same data points[19]. On the left, the curve is under-smoothed while on the right it is over-smoothed.

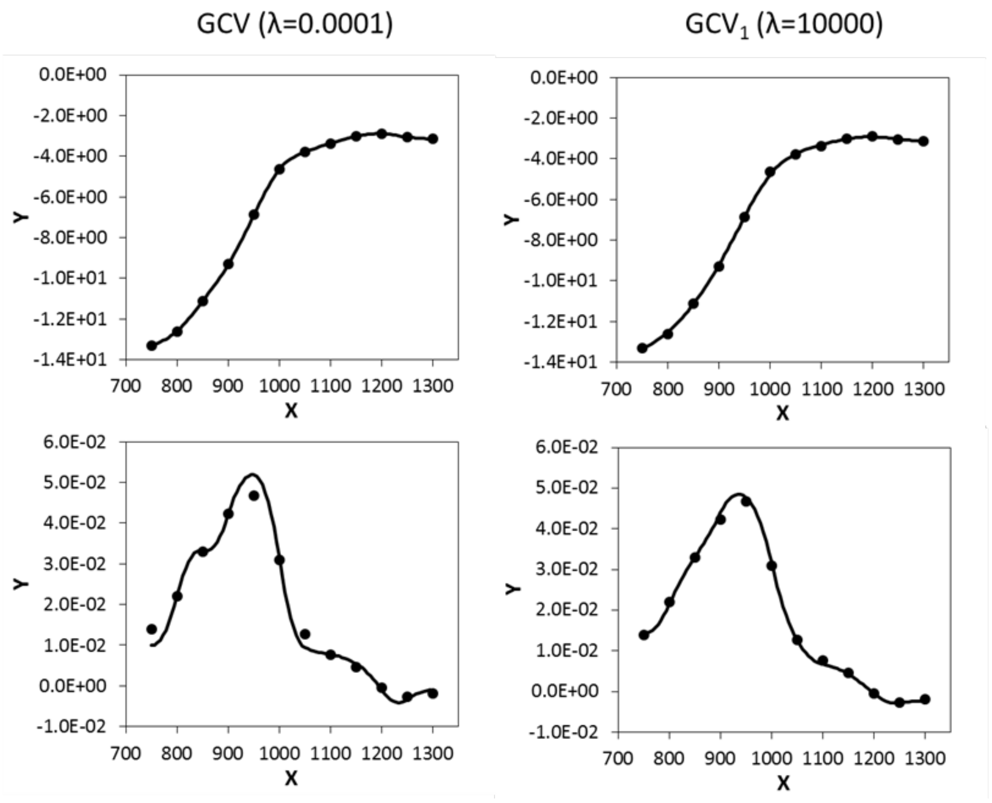


Figure 2.6: Smoothing splines obtained with different values of λ[2]

Dissimilarity measures

To assess the difference between the two functional estimates f(x) and g(x), which we will henceforth refer to as f and g (where f is the curve estimated from the experimental data and g the curve obtained from the model), the four dissimilarity measures will be defined[2].

Given the following definitions:

- f' and g' are, respectively, the first derivatives of f and g
- D is the intersection of the domains of f and g
- $\|h\|$ is the norm of a generic curve in the L2 space:

$$\|h\| = \sqrt{\int_D h(x)^2 dx} \tag{2.9}$$

We can now define the four dissimilarity measures:

$$d^0_{L_2}(f,g) = \frac{1}{1+\frac{\|f-g\|}{D}} \in (0,1) \tag{2.10}$$

$$d^1_{L_2}(f,g) = \frac{1}{1+\frac{\|f'-g'\|}{D}} \in (0,1) \tag{2.11}$$

$$d^0_{Pe}(f,g) = 1 - \frac{1}{2}\left\|\frac{f}{\|f\|} - \frac{g}{\|g\|}\right\| \in (0,1) \tag{2.12}$$

$$d^1_{Pe}(f,g) = 1 - \frac{1}{2}\left\|\frac{f'}{\|f'\|} - \frac{g'}{\|g'\|}\right\| \in (0,1) \tag{2.13}$$

We can immediately see that for all 4 measurements, the minimum value is 0, which indicates the minimum dissimilarity between the two curves.

The dissimilarity measure $d^0_{L_2}$ is the generalization to the continuous case of the SSE (Equation (2.4)): the usage of Equation (2.9) allows the evaluation of the difference between the areas of the two curves instead of the sum of the point differences.

Measure $d^1_{L_2}$ assumes a minimum value, corresponding to two perfectly similar curves, at functions that differ from each other only by a vertical translation:

$$d^1_{L_2}(f, f+a) = 0 \quad \forall a \in \mathbb{R}$$

Thus, Equation (2.11) is invariant to vertical translations and is able to assess to which extent two functions have similar slope.

d^0_{Pe} is obtained from Pearson's correlation index, which measures the correlation between two functions that assesses the similarity of their shape and considers as similar two functions that differ only by a vertical dilation:

$$d^0_{Pe}(f, f \times a) = 0 \quad \forall a \in \mathbb{R}$$

d^0_{Pe} considers as perfectly similar two curves which differ only by a vertical affine transformation (translation and dilation):

$$d^1_{Pe}\left(f, f \times a + b\right) = 0 \quad \forall a \in \mathbb{R} \quad \forall b \in \mathbb{R}$$

The four dissimilarity measures take into account the difference in the curves along the vertical direction. To obtain an evaluation of the difference in the horizontal direction, the curves must be aligned by calculating the horizontal shift between f and g, defined as the term that maximises the sum of Equations (2.10) to (2.13):

$$\delta = argmax_{\delta}(d^0_{L_2} + d^1_{L_2} + d^0_{Pe} + d^1_{Pe})$$

Then the shift S is defined, which measures the dissimilarity in terms of the horizontal shift between the two curves:

$$S = max(1 - \frac{\delta}{D}, 0) \in (0, 1)$$

After horizontal alignment, the dissimilarity measures between the functions are calculated again.

Finally, the global index M that evaluates the goodness-of-fit of the model is calculated by the averaged sum of the indices in Equations (2.10) to (2.13) recalculated after the shift, and the shift S, multiplied by two as it takes into account both the left and right horizontal shifts.

$$CM = \frac{d^0_{L2,shift} + d^1_{L2,shift} + d^0_{Pe,shift} + d^1_{Pe,shift} + 2S}{6} \tag{2.14}$$

The effectiveness of such an approach based on dissimilarity indices is evident if we compare the values obtained from Figure 2.1 models via EFV (Equation (2.1)) with those obtained via Equations (2.10) to (2.13).

The main difference in Table 2.1 is that although $d^0_{L_2}$ recognises M_3 as the most accurate model in reproducing the experimental observations, as well as EFV, $d^1_{L_2}$ is able to catch the clear superiority of the M_1 model in reproducing the slope of the experimental data in contrast to M_2 and M_3.

Obtaining from Equation (2.14) the final dimensionless index, however, requires manipulating the measurements several times with normalisation, shifting and averaging. Operations performed within this stage of the pipeline could lead to a loss of generality.

The expression Without Loss Of Generality (WLOG) [18] is frequently used in mathematics: it is used to indicate that the assumption that follows has been chosen arbitrarily, restricting the premises to a particular case but still retaining their validity.

In this case, however, the term will be used in a broader sense: what these operations may introduce, in essence, is a loss of generality in their ability to measure dissimilarity between curves. To better clarify the concept, a practical example can be given.

Let us assume that a modeller needs to validate, via Curve Matching, the performance of two models against experimental data, and that the functional estimates of the two models are visually similar.

Yet, after performing the comparison, the modeller discovers that both curves have slightly different scores against the experimental data, and that the values of the dissimilarity indices differ little from each other.

Since the modeller's purpose is to recalibrate the models to ensure that they are able to simulate the behaviour of the experimental data, in this scenario he may be in trouble because he is unable to ascertain, through the score and the various measurements, which model specifications need to be recalibrated and how much they need to be recalibrated.

Nevertheless, this is a very general case; in the following work we will assess the uncertainty introduced by a possible loss of generality in the calculation of the dissimilarity measurements and the final score.

Original	$d^0_{L_2}$	$d^1_{L_2}$	d^0_{Pe}	d^1_{Pe}	EFV
M_1	0.54	0.13	0.01	0.02	213
M_2	0.42	4.30	0.07	0.00	203
M_3	0.37	2.60	0.02	0.39	168

Table 2.1: Non-normalized dissimilarity indices for the example of Figure 2.1 [2]

Bootstrapping procedure

In the last stage of the Curve Matching pipeline, before returning the final similarity index, further data manipulation by bootstrapping takes place.

This procedure is necessary because the reliability of the index in Equation (2.14) is affected by the uncertainty of the experimental measurements: for this reason, an attempt must be made to take into account the impact that experimental uncertainty has on the final evaluation.

For this reason, the bootstrapping procedure, as described by the authors of [14], is used: taking into account the uncertainty of each measurement, a number of random values are generated for each point with a mean corresponding to the value of the measurement and a standard deviation equal to the uncertainty of the point itself: several tens of points are needed to achieve statistical significance.

Once these random values are obtained, a number of curves equal to the number of values obtained are derived (Figure 2.7), and for each value the CM index (Equation (2.14)) is calculated.

The average of the indices returns the final assessment of the goodness of fit of the model against the experimental values and the range of uncertainty of that result.

Bootstrapping is a widely used technique in statistical inference [19], numerous variants of which have been developed because of its ease of application in numerous mathematical fields.

The main idea of this technique is to estimate properties of the statistical distribution of data by measuring these properties from sampling an approximate distribution (often the standard one is the distribution obtained from empirical measurement).

In this case, for each sample from the experiments, a distribution is constructed having as the mean the value of the experimental measurement and as the standard deviation the value of the uncertainty (when available).

The goal is thus to generate a satisfactory number of samples in such a way as to reduce the influence of the uncertainty inherent in the experimental data, resulting in different functional estimates that precisely take into account that each of the points can be generated by a statistical process with certain properties.

In an interesting paper [20] the author analyzes the validity of approximations of functional estimates by bootstrapping. In fact, research on the validity, at least asymptotically, of approximations by bootstrapping dates back to the inception of bootstrapping as a popular resampling technique.

Often, the validity of distributions obtained from bootstrap counterparts is proven by showing that the distance between the statistical distributions of bootstrap and experimental samples tends to zero. In principle, we can say that bootstrapping generates reliable approximations of sample distributions, however, as already mentioned this technique is used in numerous areas and its reliability is guaranteed by theory, but in a very general way.

Some common disadvantages of bootstrapping, in fact, are important in the current context. The quality of this technique depends very much on the estimator used, and in our case the mean and standard deviation estimators are uncertain values. In addition, the bootstrapping result is highly dependent on the representative sample: again, an uncertain measurement.

Another important aspect to consider in bootstrapping samples is the assumption of independence of observations. This property is very strong in that if two observations are statistically independent, we cannot learn about one by observing the other: in a nutshell, it assures us that the extracted samples can be considered as data from stand-alone experiments, independent of each other. In our case, the data are obtained by running experiments, by hypothesis independent of each other. This guarantees independence among the experimental data and thus independence among the bootstrap samples obtained from them.

Because of this property, bootstrapping has gained popularity especially in the construction of confidence intervals, which in our application represent the probability that the experimental measurement is within a certain range, considering its inherent uncertainty. For example, in Figure 2.7 we can see that the samples extracted by bootstrapping stay within a certain range, established by the uncertainty with which the data were sampled. In this way, the confidence intervals certify that the actual experimental observations lie within a range, which in the case of Figure 2.7 is $[x - \mu, x + \mu]$, where x represents the experimental observation and μ the uncertainty associated with it.

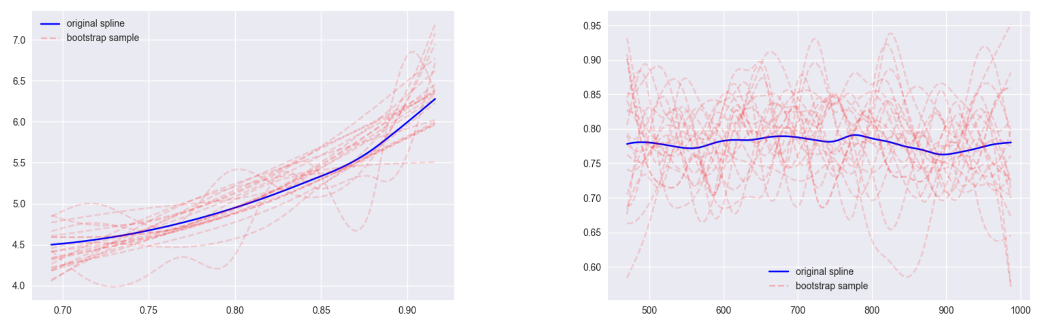
Although bootstrapping is a proven procedure in many areas, its use could potentially negatively affect the final CM index: it is in fact the result of averaging measured indices against experimental data on bootstrapped-generated curves, which for the reasons just explained could misrepresent the underlying process of data generation.

In curves with a regular trend as in Figure 2.7a, resampling by bootstrapping returns curves with a similar trend but with a larger point-to-point distance, while in curves estimated from experimental data with a highly variable trend as in Figure 2.7b return bootstrapping also returns curves with a different trend from the original one.

In addition, the experimental data shown in Figure 2.7 have an experimental uncertainty of 10%, while in the SciExpeM [12] database there are experimental data with a significantly larger uncertainty.

The uncertainty, as expected, greatly affects the final index since clearly curves that are far from the original curve will have a negative weight on the index derived after bootstrapping.

For each case the contribution of bootstrapping has a different weight: later we will go on to specifically analyze the impact this technique has on the splines computed in the SciExpeM framework.



(a) Spline from ignition delay measurement with 10% uncertainty in a shock tube reactor with 20 bootstrap samples.

(b) Spline from jet stirred reactor measurement with 10% uncertainty in a stirred reactor with 20 bootstrap samples.

Figure 2.7: Comparison of splines obtained from experimental data with 10% uncertainty

2.2. Modelling uncertainties

In the framework just described, it is clear that validating a model is a complicated process, yet it seems that once the procedure to be followed for the specific case is established, one may be able to establish with certainty the reliability of a model. Nevertheless, there is one important aspect, namely the quality of the experimental data used, which could worsen the predictive capabilities of our model, even after a careful validation procedure.

The quality of experimental data is worsened by a lot of factors such as experimental errors, misrepresentations and lack of data about experiments themselves, such as uncertainties [12]: the latter is always present in data but often it is not reported, although it is significant.

A clear picture on the assessment of Data Quality will be presented in the next section; now we will focus on how uncertainty is represented in computational models and how validation metrics take it into account.

The issue of including uncertainty in data when validating the model, instead of a mere evaluation of the output, was raised several years ago: it was clear that somehow uncertainty must be built at the beginning of the analysis and propagated through the final result [21].

As a result, research shifted towards assessing confidence in model prediction under uncertainty, which entails its quantification in models and experimental measurements: qualitative validation metrics, based mainly on graphical comparisons, were not suitable, and so techniques based on a probabilistic approach for rigorous quantification of prediction and measurement errors and uncertainties were introduced.

In [22], the authors consider uncertainty in the experimental data explicitly, and the validity of the models according to a pass/fail criterion is attested through the Bayes factor. The latter basically assesses whether experimental observations support one model over another: if a single model is proposed, the Bayes factor allows it to be accepted or rejected.

Furthermore, recent research involving Machine Learning (ML) or Deep Learning (DL) computational models has introduced the possibility to distinguish between measurement errors within experimental data and the uncertainty that is intrinsic to the prediction model [23, 24].

In particular, the types of uncertainties present in deep learning models are analyzed in the work of Kendall and Gal [24]. These uncertainties are captured through Bayesian deep learning approaches and are of two types: aleatoric uncertainty and epistemic uncertainty. In practice, aleatoric uncertainty captures the noise of observations, that is, the uncertainty that is inherent in experimental measurements. Epistemic uncertainty, on the other hand, takes into account the uncertainty in the model parameters, i.e., ignorance about what the underlying process is in creating the collected data.

With this kind of analysis, validation metrics are expressed as probabilities by which predictions and observations agree, representing the extent of validity of the model, rather than a single pass/fail judgement.

As already mentioned, each model is validated with different techniques depending on its context and domain of interpretation, and to further broaden the scenario of validation techniques and metrics, various factors inherent to Data Quality, specifically uncertainty, contribute.

However, for the purposes of this thesis, it is not necessary to go into the details of the more advanced statistical techniques used to model uncertainty: it is interesting to have an overview of all the techniques that have been developed over the years to cope with this problem, as it highlights the lack of a general framework for establishing the reliability of a model generated from low-quality data.

In fact, the ultimate goal is to assess how Data Quality impacts the final evaluation of the validation metrics, as it is on the basis of the latter that experts assess the actual goodness of the model and, if necessary, recalibrate it.

Next, an overview of uncertainty quantification and propagation approaches will be presented. These techniques differ from those for statistical evaluation of experimental uncertainty as they exploit data from experiments and associated predictive models, instead of creating boundaries based on estimates of the underlying statistical distributions of the data.

2.2.1. Uncertainty Quantification

The problem of Uncertainty Quantification (UQ) can be approached from different points of view: so far, the two main types of uncertainties have been introduced, namely, the aleatoric and epistemic uncertainty.

However, even with a model that accurately represents the real-world phenomenon, methods for exploiting experimental data for calibration and prediction are not yet standardized. In fact, in addition to the techniques already mentioned, there are several approaches in the literature, statistical and otherwise, that combine models and experimental data from different sources to explore their information content and quantify uncertainty and its propagation.

In fact, the problem of uncertainty quantification has been addressed since the early 2000s by focusing on reducing the complexity of mathematical models, thus reducing their dependence on a large number of parameters, which precisely makes them unreliable.

Numerous approaches [25–27] address the UQ problem through optimization of surrogate models: in practice, restrictions are placed on the possible values that the mathematical model's predicted values return, and a result is satisfactory if its predictions fall within certain boundaries.

However, since the equations that constitute the model often have a solution that is difficult to compute: they are in fact approximated, and in the optimization problem, instead of using the original mathematical model, a model based on these "simplified" equations, called a surrogate model, is used.

Surrogate model-based techniques, in practice, help to select the most suitable model parameters through numerical optimization.

Another interesting approach involves data collaboration methods [28, 29], which allow consistency evaluation of a dataset for selection of the best models.

In this approach, each experiment is assigned a dataset unit defined by (d, u, M), where d is the measured value, u the reported uncertainty, and M the mathematical model of the experiment. The true value of the experimental observation y, satisfies $|d - y| \leq u$

Next, a measure of consistency is introduced, where the mismatch between model and data is minimized: a mismatch within a certain error level is acceptable and makes the dataset consistent.

The fitness of a model is assessed by incorporating each candidate model into a separate dataset: that is, for each dataset the experiments are the same, as is the associated uncertainty, only the candidate model changes. For each dataset, the consistency measure is evaluated, and the one with the greatest value of the latter is selected as the candidate model.

One remark should be made about this approach: compared to surrogate models, in Data Collaboration model selection is also made based on the uncertainty inherent in the experimental data. So, in this case, the approach to UQ considers as best the model that succeeds in obtaining better predictions, with an error no greater than the uncertainty inherent in the experimental data. Clearly, this is possible only when the uncertainties are explicitly stated in the experiments.

In [29] the authors present an approach for optimizing models through both surrogate models and Data Collaboration. The authors emphasize how important it is to incorporate uncertainty into the analysis of predictive models given in reality, just as it is impossible to have experimental observations without uncertainty, it is impossible for even the most advanced mathematical models to be free of approximations and parametrizations that introduce uncertainty.

In this paper [30] the authors compare two UQ frameworks: the first is Bound-To-Bound Data Collaboration (B2B), based on the previously mentioned Data Collaboration approach, which produces deterministic uncertainty boundaries to make predictions. The second one, on the other hand, based on Bayesian statistics, is Bayesian Calibration and Prediction (BCP), and basically like the statistical approaches observed so far, it allows to evaluate probabilistic distributions of the predicted data.

Both proposed approaches succeed in generating satisfactory uncertainty assessments by not considering the presence of bias in the model, which is a very strong assumption, as pointed out by the authors of [24]. In fact, not considering bias in the model is equivalent to not taking into account epistemic uncertainty, which as seen in [24] is a strong component of uncertainty, especially in the context of certain computational models.

Yet the domains of application of the various uncertainty quantification techniques should not be confused. The use of techniques for quantifying *aleatoric* and *epistemic* uncertainty in fact pertains to the world of computational models, finding application mainly in DL models.

Techniques such as B2B and BCP, on the other hand, see their application more focused on the computation of uncertainty in the parameters of predictive models for complex scientific experiments.

Strong assumptions such as lack of bias are due to the fact that, as stressed again, the availability of experimental data depends on a number of factors, and thus it is often difficult to have accurate information about a given real-world phenomenon available.

Furthermore, when it comes to having to mathematically abstract a complex real-world phenomenon, a major difficulty arises: more accurate and authentic models introduce more parameters.

Currently, especially in fields where the phenomena to be modeled are very complicated, the techniques used to quantify uncertainty are mixed and include the principles of Data Collaboration, surrogate models, or statistical approaches. Just as in model validation, therefore, in Uncertainty Quantification there is no well-defined approach for each specific type of need.

2.2.2. Uncertainty Propagation

It is crucial to comprehend how uncertainty propagation takes place inside the pipeline after establishing a methodology for uncertainty quantification in a challenging environment like a multistage pipeline.

It is evidently challenging to find a ready-made approach to use in this situation as well, as one must take into account a number of factors including the validity of the experimental data, the complexity of the models used, the inherent limitations in the quantification of uncertainty itself, and how this propagates from one stage of the pipeline to the next.

Furthermore, as discussed later in Section 2.4.1, the data may be subject to transformations that modify its quality and add to the uncertainty.

The problem of uncertainty propagation arises when a given quantity of interest is a function of other quantities of interest.

This would imply, for example, in the case of curve matching, being able to establish the relationship between the uncertainty in the final CM index of Equation (2.14) and the dissimilarity indices, the uncertainty of which is dependent on the functional estimations, and so on.

It must also be said that most approaches to uncertainty propagation existing in the literature focus on a probabilistic validation of predictive models (hence the return of a model reliability index), whereas in the case of Curve Matching we are interested in how the various uncertainties propagated along the multi-stage pipeline are reflected in the final index.

An interesting article from 2016 [31] discusses a multitude of approaches for model validation in different scenarios, studying the propagation of uncertainty by distinguishing between epistemic and random uncertainty. Validation is done using model reliability metric, developed by Rebba and Mahadevan ([32]. It is defined as the difference (Δ) between observed data (Y_D) and model prediction (Y_m) being less than a tolerance limit (ϵ):

$$r = Pr(-\epsilon < \epsilon), \quad \delta = Y_D - Y_m \tag{2.15}$$

The approach to the problem changes according to the characteristics of the experimental data available as input to the model, i.e. three cases are considered: (1) all experimental data are measured and reported as point data (fully characterised data), (2) some data are reported as intervals (partially characterised data) and (3) some experimental data are not measured or reported as single interval (uncharacterised data).

The data can be viewed individually and compared against a separate stochastic prediction at each input condition, or can be seen collectively. It is difficult to disentangle the contributions of aleatory and epistemic uncertainty sources to the validation outcome if the validation evaluation is performed only once over the collection of data (i.e., ensemble validation). In the latter case, the distributions of both the prediction and the observation are a result of aleatory uncertainty (input variations) and epistemic uncertainty (parameter uncertainty).

The p-box approach is one alternative for distinguishing the aleatory and epistemic components in the collective perspective of validation. Epistemic uncertainty is expressed as an interval in this treatment, whereas aleatory uncertainty is expressed using probability distributions. Because the Data Quality does not allow for point-by-point separation, this approach is especially appropriate for uncharacterized data.

When the major influence is epistemic uncertainty rather than aleatory uncertainty, comparing observations to a p-box may be ineffective because epistemic uncertainty provides a large window of acceptability for the model. Many problems have significant epistemic contributions because economic restrictions in realistic applications frequently result in sparse/inaccurate data. As a result, the model reliability strategy is directed towards epistemic uncertainty in models.

When information regarding the specific input condition associated with each data point is available (either completely or partially described data), it is recommended to conduct individual comparisons with the model reliability metric at each location, which enables aleatory and epistemic uncertainty to sources to be separated from one another.

Subsequently, the impossibility of finding a feasible approach to uncertainty propagation for probabilistic model validation in the case of complicated computational models and the use of surrogate models in these cases is emphasised. The surrogate uncertainty in a surrogate model may be easily determined from the model's covariance structure, and this uncertainty results in the model reliability metric itself being viewed as a random variable with epistemic uncertainty.

Once the model reliability metric (either a single value or a distribution) has been produced, the metric can be interpreted probabilistically, allowing the validation result to be incorporated into the predictions. With this methodology, it is therefore possible to determine the various sources of uncertainty and their propagation in a predictive model.

However, despite the analysis of various scenarios available, it is not possible to use such an approach in complex applications that require the processing of data in various steps, and thus the need for an analysis of uncertainty from other points of view emerges, starting for example with Data Quality.

Sensitivity analysis

Both sensitivity analysis and uncertainty propagation are important techniques in the field of uncertainty analysis, which is concerned with quantifying and characterizing uncertainty and variability in mathematical models and simulations[33].

Sensitivity analysis is a technique used to identify the most influential parameters or inputs in a model or simulation that affect its outputs or results. The goal of sensitivity analysis is to understand which parameters or inputs have the greatest impact on outputs so that efforts can be focused on improving the accuracy and reliability of those inputs.

Uncertainty propagation, on the other hand, involves quantifying how uncertainties in the inputs to a model or simulation propagate to the outputs. This technique is used to understand how uncertainties in the inputs affect the reliability and accuracy of the outputs and to estimate the overall uncertainty in the results.

There is a close relationship between sensitivity analysis and uncertainty propagation, as sensitivity analysis is often used as a precursor to uncertainty propagation. By identifying the most influential parameters or inputs in a model, sensitivity analysis can help prioritize efforts to reduce uncertainty and improve the reliability and accuracy of the model. These efforts may include reducing uncertainty in the most influential parameters or collecting more data to improve estimates of those parameters.

Once the most influential parameters have been identified, uncertainty propagation can be used to estimate how uncertainties in these parameters propagate through the outputs and to estimate the overall uncertainty in the results. By combining the results of sensitivity analysis and uncertainty propagation, it is possible to gain a better understanding of the factors that contribute most to uncertainty in the model and to develop strategies for improving the accuracy and reliability of the results.

In the context of sensitivity analysis and uncertainty propagation, the Spearman coefficient can be used to identify how the ranking or order of inputs or parameters in a model correlates with the ranking or order of outputs or outcomes[34]. This can help identify which inputs or parameters are most strongly associated with the outputs and which have little or no influence on the results.

The Spearman coefficient, also known as Spearman's rank correlation coefficient, or Spearman's ρ_S, is a measure of the strength of the association between two variables. It measures the degree to which the ranks of two variables are correlated with each other, rather than their actual values. It is defined as:

$$\rho_S = \frac{\sum_i (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\sum_i (r_i - \bar{r})^2}\sqrt{\sum_i (s_i - \bar{s})^2}} \tag{2.16}$$

Where r and s are the ranks of the variables in analysis : in statistics, ranking is the data transformation in which numerical or ordinal values are replaced by their rank when the data are sorted.

Intuitively, the Spearman correlation between two variables will be high if the observations have a similar (or identical for a correlation of 1) rank (i.e., the relative position label of the observations within the variable: 1st, 2nd, 3rd, etc.) between the two variables, and low when observations have a dissimilar (or completely opposite for a correlation of -1) rank between the two variables.

In addition, the Spearman coefficient can be used to assess the robustness of sensitivity analysis results. When the Spearman coefficient is high, it indicates a strong correlation between the ranks of the inputs and the ranks of the outputs, suggesting that the results of the sensitivity analysis are robust and reliable. On the other hand, if the Spearman coefficient is low, it indicates that the ranks of the inputs are not strongly correlated with the ranks of the outputs, indicating that the results of the sensitivity analysis may be less reliable.

2.3. Data Quality

The impact of Data Quality, from a research perspective, has been studied in various areas including statistics, management, and computer science. The earliest research efforts on the impact that Data Quality has in informatics date back to the 1990s: as the years passed and the development of increasingly advanced data-driven models, interest in this area soared as a strong correlation was discovered between the development of successful models and the quality of the data used.

Real-world data are often incorrect, incomplete, and contain many errors and cause a lot of damage to the organizations that use them and not only in economic terms: Figure 2.8, shows all the costs related to business processes and data management due to poor Data Quality.

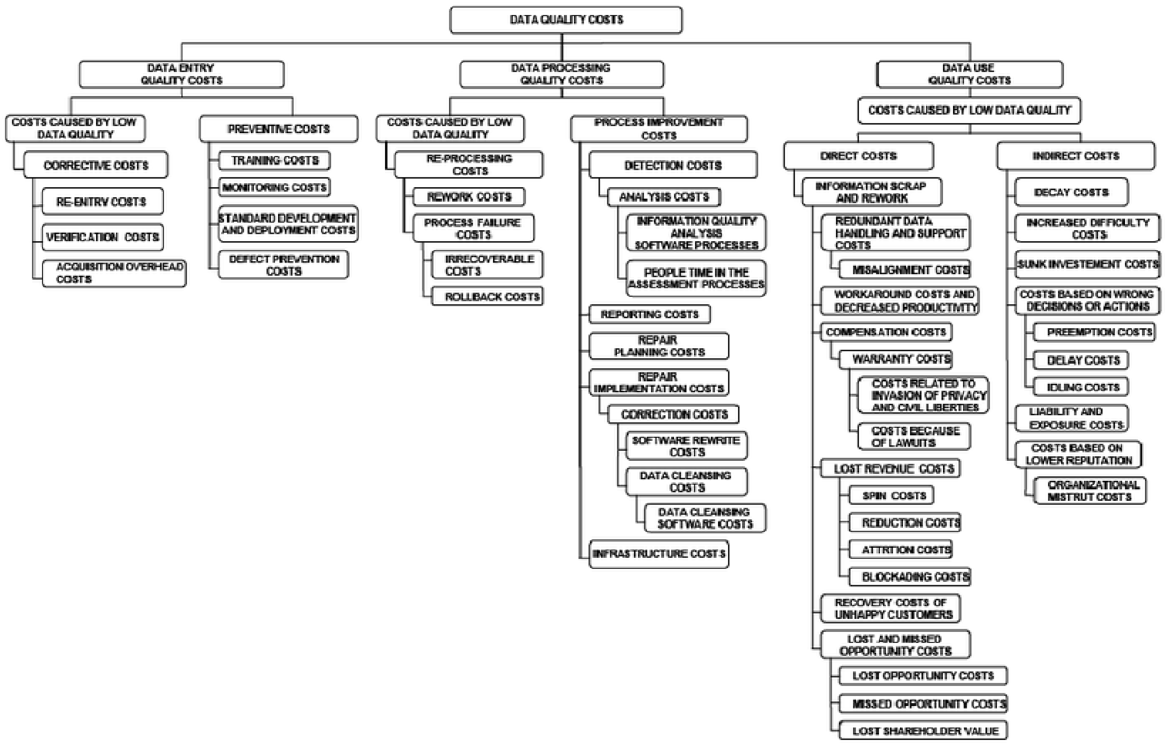

DATA QUALITY COSTS
DATA ENTRY QUALITY COSTS
DATA PROCESSING QUALITY COSTS
DATA USE QUALITY COSTS
COSTS CAUSED BY LOW DATA QUALITY
PREVENTIVE COSTS
CORRECTIVE COSTS
RE-ENTRY COSTS
VERIFICATION COSTS
ACQUISITION OVERHEAD COSTS
TRAINING COSTS
MONITORING COSTS
STANDARD DEVELOPMENT AND DEPLOYMENT COSTS
DEFECT PREVENTION COSTS
COSTS CAUSED BY LOW DATA QUALITY
RE-PROCESSING COSTS
REWORK COSTS
PROCESS FAILURE COSTS
IRRECOVERABLE COSTS
ROLLBACK COSTS
PROCESS IMPROVEMENT COSTS
DETECTION COSTS
ANALYSIS COSTS
INFORMATION QUALITY ANALYSIS SOFTWARE PROCESSES
PEOPLE TIME IN THE ASSESSMENT PROCESSES
REPORTING COSTS
REPAIR PLANNING COSTS
REPAIR IMPLEMENTATION COSTS
CORRECTION COSTS
SOFTWARE REWRITE COSTS
DATA CLEANSING COSTS
DATA CLEANSING SOFTWARE COSTS
INFRASTRUCTURE COSTS
COSTS CAUSED BY LOW DATA QUALITY
DIRECT COSTS
INFORMATION SCRAP AND REWORK
REDUNDANT DATA HANDLING AND SUPPORT COSTS
MISALIGNMENT COSTS
WORKAROUND COSTS AND DECREASED PRODUCTIVITY
COMPENSATION COSTS
WARRANTY COSTS
COSTS RELATED TO INVASION OF PRIVACY AND CIVIL LIBERTIES
COSTS BECAUSE OF LAWUITS
LOST REVENUE COSTS
SPIN COSTS
REDUCTION COSTS
ATTRTION COSTS
BLOCKADING COSTS
RECOVERY COSTS OF UNHAPPY CUSTOMERS
LOST AND MISSED OPPORTUNITY COSTS
LOST OPPORTUNITY COSTS
MISSED OPPORTUNITY COSTS
LOST SHAREHOLDER VALUE
INDIRECT COSTS
DECAY COSTS
INCREASED DIFFICULTY COSTS
SUNK INVESTEMENT COSTS
COSTS BASED ON WRONG DECISIONS OR ACTIONS
PREEMPTION COSTS
DELAY COSTS
IDLING COSTS
LIABILITY AND EXPOSURE COSTS
COSTS BASED ON LOWER REPUTATION
ORGANIZATIONAL MISTRUT COSTS

Figure 2.8: Classification of costs of poor Data Quality[35].

Several factors can result in poor quality.

- Historical changes: over time, the significance of data may change.
- Data usage: the method used to collect the data will determine its significance.
- Corporate mergers: data integration may present certain challenges
- Privacy: since data are subject to privacy laws, it is challenging to discover incorrect data and establish their true database.
- Data enrichment: adding external data to internal data could be risky.

In fact, a common problem in Data Quality concerns that poor quality generates the so-called Garbage-In-Garbage-Out (GIGO) phenomenon: no matter how perfect it may be, work done on low-quality data will return a low-quality result.

This phenomenon makes clear, then, the connection between poor Data Quality and poor Quality of Service (QoS) in data-driven services (especially in ML-based services): as pointed out by the authors of [36], the value obtainable from services depends on the quality of the data interchanged by the services themselves.

The quality of a service, therefore, is affected by the knowledge on which it is based, which depends on the data underlying the service.

It is natural, then, to note that there is a close relationship between Data Quality and uncertainty in the data: both concepts are fundamental in the scientific community, and as mentioned earlier the rise of increasingly data intensive models has necessitated a deeper analysis of the quality of the data used to produce quality services.

This topic is explored more in depth in a report from the Tenth World Congress of Chemical Engineering in Barcelona [37], where issues related to Data Quality and uncertainty propagation were explored in a discussion led by academic and industrial experts, seeking answers together with the audience in three major areas of interest: data acquisition and evaluation of experimental uncertainties, tools for reconciling data to improve their quality, and the impact of data uncertainty at the end of the process.

Below are some key points from the discussion.

1. The quality of experimental data does not always meet the desired standards, and this can be attributed to several factors, such as:
   (a) The tradition of researching high-quality measurements often lost, not yet acquired, or compromised.
   (b) The exponential growth of data, connected with the emergence of new faces in the scientific field with little practice in conducting research, resulting in the publication of low-quality literature in scientific journals
2. Predictive and theoretical models are robust against errors in their domain of applicability; however, they may exhibit inpredictable behavior. Since they are based on experimental data, the quality of the underlying data commands the reliability of these models (consistent with the GIGO phenomenon).
3. Data Quality is closely associated with the person and human factor.
4. Process engineers who use process simulators carry responsibility toward the final quality of the design: however, many participants in the discussion stated that they were unaware of the quality of the data used. Therefore, a proper assessment of the quality of the data used as input is much in demand.

2.3.1. Data Quality Dimensions

The quality of data is closely related to its ability to represent the real world, but it also depends on the purpose for which it is used. In fact, a very common definition of Data

Quality is "fitness-for-use," meaning that the quality of data is measured by how well it is fit for the purpose for which it is used.

However, as pointed out by the authors of [38], this could lead to the conclusion that no objective assessment of data is possible: however, depending on the application domain of interest to the data user, it makes sense to give a quantitative assessment of certain dimensions of Data Quality of relative importance to the context in which they are used.

Beware, nevertheless, since Data Quality dimensions can refer either to the extension of the data, i.e., its value, or to its intension, i.e., its schema, and both types of dimensions are generally defined qualitatively: in fact, to give a quantitative assessment of them, one or more metrics must be associated with the specific dimension of interest. Indeed, in the analysis of this paper, certain metrics will be associated to measure the Data Quality dimensions of accuracy, completeness, and consistency.

Given the renewed importance of Data Quality assessment in many fields where data-driven decisions are critical, Data Quality research in recent years has defined numerous dimensions and related metrics for Data Quality assessment: a big step forward has been made by the study [39], which developed a framework for defining Data Quality dimensions and categorizing them through two direct surveys of data customers, following the fitness-for-use philosophy: after the first survey, as many as 179 Data Quality dimensions were identified, while against the second survey, 15 dimensions were selected to be included in the four categories that represent the most important characteristics of a data source for a customer.

In Figure 2.9 below, the division of the various dimensions into the four categories.

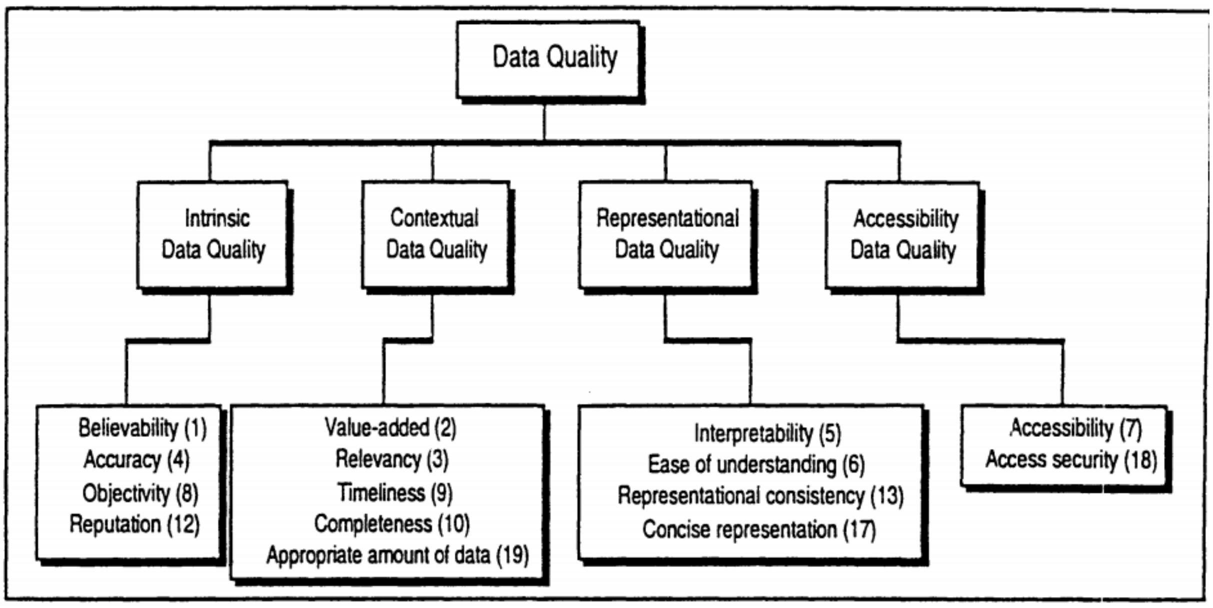


Figure 2.9: DQ categories and dimensions [39]

In the Approaches to the Definition of Data Quality Dimensions section of [38], the authors propose three main approaches for defining a set of definitions of Data Quality dimensions: the theoretical, the empirical, and the intuitive.

Next, this is followed by the definitions of categories and quality dimensions according to the empirical approach (based on the work of Wang and Strong [39]), where a two-level classification is proposed where for each of the categories is further deepened into a different number of dimensions:

- Intrinstic Data Quality captures the quality that the data itself has, e.g., accuracy is a quality that is intrinsic to the data.
- Contextual Data Quality considers the context where the data are used: completeness for example is closely related to the context of the task at hand.
- Representational Data Quality captures aspects related to the quality of data representation, such as interpretability
- Accessibility Data Quality relates to the accessibility of the data and a nonfunctional property of the data, namely the level of security

Furthermore, Table 2.2 shows the categories, associated dimensions and their definition.

In the application of the methodology proposed in this thesis, which concerns an environment for the development of data-driven simulation models, it is necessary to define the dimensions of accuracy, completeness and consistency since they are the most widely used across different domains and return a good assessment of Data Quality [12].

The use of these three dimensions is also due to the fact that defining metrics for their evaluation will allow to assess the impact that Data Quality has in propagating uncertainty in a data transformation pipeline.

Category	Dimension	Definition
Intrinsic	Believability	data are accepted or regarded as true, real and credible
	Accuracy	data are correct, reliable and certified free of error
	Objectivity	data are unbiased and impartial
	Reputation	data are trusted or highly regarded in terms of their source and content
Contextual	Value-added	data are beneficial and provide advantages for their use
	Relevancy	data are applicable and useful for the task at hand
	Timeliness	the age of the data is appropriate for the task at hand
	Completeness	data are trusted or highly regarded in terms of their source and content data are of sufficient depth, breadth, and scope for the task at hand
	Appropriate amount of data	the quantity or volume of available data is appropriate
Representational	Intepretability	data are in appropriate language and unit and the data definitions are clear
	Ease of under standing	data are clear without ambiguity and eas ily comprehended
	Representational consistency	data are always presented in the same for mat and are compatible with the previous data
	Concise represen tation	data are compactly represented without being overwhelmed
Accessibility	Accessibility	data are available or easily and quickly re- trieved
	Access security	access to data can be restricted and hence kept secure

Table 2.2: Empirical approach categories and dimensions

Accuracy

Accuracy is defined as the closeness of a v value and a v' value, considered the correct representation of the real-world phenomenon that v aims to represent.

There are two types of accuracy, namely syntactic accuracy and semantic accuracy. Syntactic accuracy is the closeness of a value v to the elements of the corresponding domain of definition D: in this case we are not interested in comparing v with a value v', but in checking that v is one of the possible values in D.

Syntactic accuracy is measured by comparison functions, which evaluate the distance between v and the elements in D. An example of a comparison function is the edit distance, which takes into account the minimum number of insertions, deletions, and substitutions of characters to convert a string s to a string s'.

Semantic accuracy is the closeness of v to the true value v': it coincides with the concept of correctness. Unlike syntactic accuracy, to measure the semantic accuracy of a value v, the corresponding true value must be known, or at least possible to infer what it is, to determine whether or not v is the true value.

In our case, we will consider semantic accuracy since it is strongly correlated with the presence of uncertainty in measurements: the basic idea is to consider the value of an experimental measurement as the true real-world value, and to evaluate how the CM pipeline presented in Figure 2.2 responds to different degrees of uncertainty injected into the data in the form of perturbations.

Completeness

As in [38], authors define completeness as "the extent to which data are of sufficient breadth, depth, and scope for the task at hand."

Usually this concept is much used in relational models, where the completeness of a table represents the extent to which the table represents the corresponding real world.

Completeness in relational models can be characterized by the presence/absence of null values and the validity of one of two assumptions called Open World Assumption (OWA) and Closed World Assumption (CWA).

The CWA says that only the values present in the relational table are considered true, so what is not defined is considered as nonexistent, while in the OWA neither the truth nor falsity of values not represented in the table can be ascertained.

Particularly interesting for the methodology applied later is the metric used to evaluate the completeness of a null-valued model with OWA.

Given a relation r, the reference relation of r, called ref(r), is the relation that represents the real-world objects that constitute the real extension of the schema, and completeness is calculated as:

$$C(r) = \frac{|r|}{|ref(r)|} \tag{2.17}$$

Following this approach, we will evaluate the impact of completeness on the CM validation method by considering the set of experimental measurements for an experiment as ref(r) in Equation (2.17).

In this way, a completeness of 100% is equivalent to having all the experimental data available: by removing measurements from the set as we go along, it will be possible to observe the extent to which a more or less complete data set affects the validation of the model.

Consistency

The consistency dimension captures the violation of semantic rules defined on in set of data objects, where the objects can be, for example, relational table tuples or records in a file.

As in relational theory, integrity constraints are the instantiation of these semantic rules, while in statistics, data edits are another example of semantic rules that enable consistency checking.

Integrity constraints are properties that must be satisfied by all instances of a database. There are two main types of integrity constraint.

- Intrarelation constraints, which involve individual attributes or multiple attributes of a relation, for example: the attribute describing a person's age must be between 0 and 120.
- Interrelation constraints: involving attributes belonging to more than one relation.

Most integrity constraints are dependencies. The main ones are:

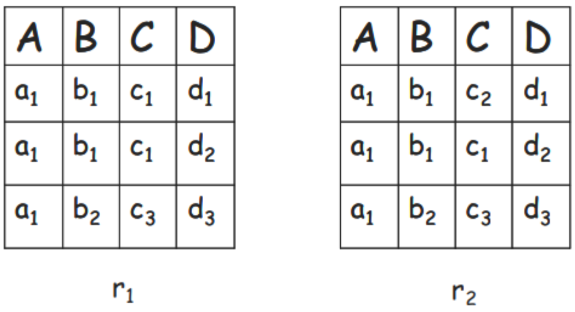
- Key Dependency. Given an instance of a relation r, defined over a set of attributes, we say that for a subset K of the attributes holds a key dependency if no two rows have the same values in K.
- Inclusion Dependency. An inclusion dependency on a relational instance r says that some columns of r are contained in other columns of r or in instances of another

relational instance s.

- Functional Dependency (FD). Given a relational instance r, let X and Y be two nonempty sets of attributes in r. r satisfies the functional dependency $X \rightarrow Y$ if the following holds:

$$if\ t_1.X = t_2.X\ \ then\ \ t_1.Y = t_2.Y \quad \forall t_1, t_2 \in X,\ \forall t1, t_2 \in Y$$

  In Figure 2.10 this concept is better clarified: the table r_1 is an example where the FD $AB \rightarrow C$ holds, while in r_2 the same FD doesn't hold.



A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_1	b_2	c_3	d_3

r_1

A	B	C	D
a_1	b_1	c_2	d_1
a_1	b_1	c_1	d_2
a_1	b_2	c_3	d_3

r_2

Figure 2.10: Example of functional dependencies [38]

The consistency dimension can be analyzed from multiple perspectives: in our case we will define a metric based on a particular type of functional dependency constructed ad-hoc for evaluating the impact of this dimension on Curve Matching.

In short, we are going to analyze a dataset of measurements from the same experiment: this dataset will be 100% consistent when the measurements all fall within a given range of values.

Let us imagine that table r_1 in Figure 2.10 is the reference dataset, where column A contains the reference values for the experiment measurements, and columns B,C,D contain other measurements. Thus, for each row we want b_1, c_1, d_1 to be contained in the range $[a_1 - \epsilon, a_1 + \epsilon]$, where ϵ is a value that will be defined for each experiment.

2.4. Data Pipeline

When we talk about pipelines in computer science, we can refer to several things. However, referring precisely to the physical meaning of pipeline, a given pipeline is basically a series of elements that process data connected in series, such that the output of one element is the input of the next.

In recent years, however, with the advent of Big Data, data analysis pipelines have emerged, which have been given this very specific name because almost all applications in which data is involved and processes data follows five major stages [40]:

- Acquisition and recording: the first stage in which data is collected from the various sources.
- Information Extraction and Cleaning: in this stage the data is manipulated so that it is ready to be analyzed, since it is often not in the right format or has other problems.
- Data Aggregation, Integration and Representation: given the heterogeneity of the streams from which the data comes, it is often not enough simply to have the data itself. One must perform a transformation of the data to make it homogeneous and effectively usable in the next steps. This problem, as pointed out by the authors of [1] is especially relevant in the field of natural sciences, where the result of data analysis is strongly influenced by the volume, sparsity and imprecision in data sources.
- Query Processing, Data Modeling and Analysis: at this stage the actual data analysis takes place, which can be carried out by different types of algorithms and vary depending on the domain of interest and the type of data being analyzed (very common is the use of Data Mining algorithms in Big Data analysis, for example). These algorithms aim to analyze data and find a model that simulates the generation process underlying it.
- Interpretation: in the last stage, the interpretation of the analysis performed by the pipeline takes place. In fact, decisions often depend on these analyses, so the interpretation of the result is crucial. Showing the results is rarely enough to meaningfully interpret the data, so additional information (such as provenance) or graphical resources that enrich the simple numerical result are often provided along with them.

These steps obviously embrace in their generality all the possible applications of a data analysis pipeline, applicable to a multitude of problems: as far as the application in analysis is concerned, the steps recognisable from Figure 2.2 are certainly those of data acquisition from the experimental database, data modelling and analysis in the step of constructing functional estimates and in bootstrapping, and finally the interpretation of the final result by means of the Curve Matching index.

What emerges is that data pipelines are essential components in modern data analysis workflows. Moreover, not all pipelines handle Big Data, so even the operations listed above can actually be further abstracted into two macro phases: one for data preprocessing and one for actual analysis. This theory is also supported by [3], where the authors propose a framework to support pipelines for data preparation: here, precisely, the macro-distinction in the two categories is emphasized.

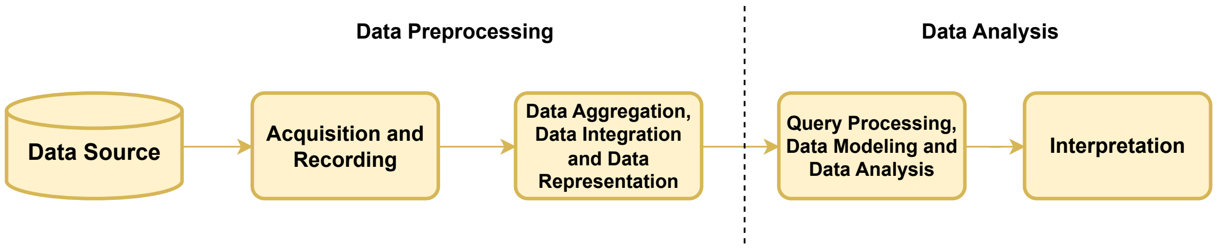

Data Preprocessing
Data Analysis
Data Source
Acquisition and Recording
Data Aggregation, Data Integration and Data Representation
Query Processing, Data Modeling and Data Analysis
Interpretation

Figure 2.11: A representation of a Data Pipeline, where the two macro-steps of Data Preprocessing and Data Analysis are highlighted.

The data preprocessing step, regardless of the data handled, is always necessary since raw data are often messy and unsuitable for analysis. Data cleaning, feature engineering, and data transformation techniques are used to transform raw data into a more appropriate format for analysis. This step could also include data integration, where data from different sources are combined to create a unified dataset.

The second macro step is the actual analysis of the data, which is often done automatically using algorithms or sometimes the data are analyzed by experts who draw conclusions. Usually algorithms process the prepared data and produce results that are easy to visualize and interpret. The output is often in the form of a visualization, report, or predictive model: in fact, this step might also include statistical analysis, Machine Learning, or Deep Learning.

After categorizing the macro steps of a given pipeline, it is therefore important to evaluate the results to ensure the validity of the analysis. An excellent example comes from [41], where the authors develop a social media analysis pipeline for automatic extraction of information to characterize, highlight and react to emergencies.

First, here again we can see that the macro steps in the pipeline are those of data preprocessing and data analysis. What is interesting is the validation process, which results in a mix of different methods namely individual, group, computation-based and validation datasets.

Individual validation involves having experts evaluate the output of the pipeline to assess its accuracy and validity. This method is often used for qualitative analysis, where the output needs to be interpreted and evaluated by humans.

Group validation involves having multiple individuals vote on the output of the pipeline to determine the consensus view. This method is often used in crowd-based approaches, where a large group of individuals contribute to the analysis.

Computation-based validation involves using automated methods to assess the performance of the pipeline. These methods are particularly useful for quantitative analysis, where the output can be evaluated based on specific metrics or criteria (as it happens in CM, described in Section 2.1.3).

Finally, validation datasets are external sources of data used to validate the output of the pipeline. These datasets can be used to compare the output of the pipeline with known ground truth values.

The authors of [41] use these techniques to validate the results of the methodology applied on three different case studies, managing to include several dimensions of Data Quality in the analysis using computation-based validation. This results in an efficient validation process.

It emerges from these observations, given the transformational nature of the pipeline, of a rigorous validation procedure especially when dealing with raw data. As emphasized several times, the transformations that occur in a pipeline impact the Data Quality of the data used and may lead to a more or less valid final result.

2.4.1. Data Transformations

When carrying out any kind of analysis, often the biggest stumbling block between a scientist and a satisfactory result is the quality of the data at hand, so performing a series of operations to improve the quality of the data has become an almost obligatory practice nowadays.

Because data is frequently collected from various sources that are not always accurate and come in a variety of formats, and there may be issues owing to human error, measurement instrument limitations, or defects in the data collection method, preparation must be achieved in a rigid manner and can take a long time.

Beware, preparation operations do not necessarily require a transformation of a data, but may be simple checks to establish the reliability of the data set at hand. In fact, the preparation of data for analysis includes data control operations, statistical operations e.g. to eliminate outliers, and actual transformation operations, which require modifications to the data in order to be suitable for their purpose.

Obviously, the final result of the analysis will depend on the data that has undergone transformations, and a doubt naturally arises: can processing the data and transforming it negatively affect the final result? Finding a sure answer is difficult, however, in order to use scientific simulation models, as in the case of Curve Matching, processing the data is necessary as it is fed to a parametric model that accepts only correctly formatted inputs.

Furthermore, for the reasons mentioned at the beginning, it is very common for errors in data collection to be carried over into the analysis and result in an incorrect representation of the underlying physical phenomenon.

Precisely for this reason, one of the tasks to be performed before analysing the data is the removal of outliers: these data have the characteristic of deviating significantly from the other observations and therefore carry more weight than them. Their removal, however, is not easy, as in the case of complex phenomena, they might contain truthful information and therefore need to be handled carefully.

Another very common problem is that of data representation, especially when it comes to experimental measurements. Due to various factors, such as the scale of the measurements, it can be difficult to capture the nature of the data at hand.

The most common transformation of all is the logarithmic scale transformation, which is of great importance in statistics, to recognise the underlying distribution of the data. Sometimes, it is also useful for comparing different data with each other or for capturing trends that would be impossible to see at the original scale.

Like the logarithmic scale transformation, mathematical operations performed in a similar manner on data can be trigonometric transformations, square root, inverse or normalisation. As mentioned earlier, these transformations are often useful in statistics because in many procedures, variables are assumed to be normally distributed and by manipulating the data mathematically, they are brought back to a normal distribution.

However, it is rare in the scientific world for experimental measurements of a phenomenon to assume a normal distribution. In fact, as mentioned earlier, it can happen that errors in data collection or missing values, as well as the presence of outliers or the nature of the data itself, can lead to a non-normal distribution.

In a framework such as Curve Matching, data are subjected to a series of mathematical operations such as the addition of constants, multiplications, logarithmic transformations and square roots that affect the result due to their inherent nature of changing the nature of the data.

Let us take for example three transformations mentioned in this section and see what their characteristics are and the problems they can bring [42].

- Square root transformation. Each value is squared. However, as it is not possible to calculate the square root of a negative integer, if a variable has negative values, it is necessary to introduce a constant to raise the minimum value of the distribution above 0, preferably to 1, because numbers from 1 upwards behave differently to numbers between 0 and 0.99. The square root of numbers greater than 1 is always smaller, 1 and 0 remain constant, while numbers between 0 and 1 are always larger. Consequently, if you apply a square root to a continuous variable with values between 0 and 1 and greater than 1, you treat some integers differently from others, which is probably not desired in most circumstances.

- Logarithmic transformations. Logarithmic transformations are a type of transformation, not a particular transformation. A logarithm is the power to which a base number must be increased to obtain the original number. For log transformations, base 10 is not the sole option. Another popular choice is the natural logarithm, where the base is Euler's number. Because the logarithm of any negative integer or number less than one is undefined, if a variable contains values less than one, a constant must be introduced to move the distribution's minimum value, preferably to one.

- Inverse transformation. Taking the inverse of a number (x) is equivalent to calculating 1/x. This increases the size of extremely small numbers and decreases the size of very large numbers. This change reverses the order of the scores. Therefore, before using an inverse transformation, it is necessary to reflect or invert the distribution. To reflect, one multiplies a variable by -1, then adds a constant to the distribution to restore the minimum value above 1. At the end of the inverse transformation, the ordering of the values will be similar to that of the original data.

What these transformations do, in essence, is alter the distances between the points: this is what creates problems in the interpretation of the data. If the operation is performed correctly, all data points remain in the same relative order as before the transformation. However, this may be undesirable as the variables become more complex to interpret due to the curvilinear nature of the transformations.

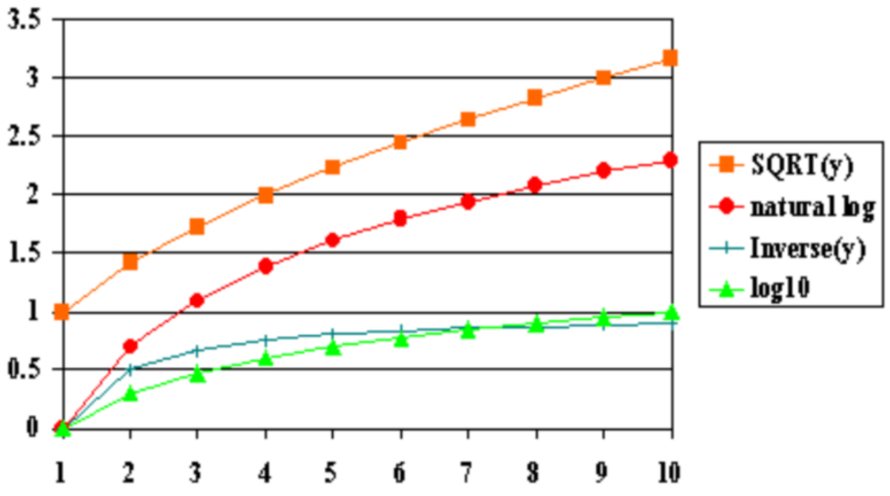


Figure 2.12: The effect of transformations on variables [42].

Normalization

Normalization is a common technique used in data preprocessing to scale features to a uniform range. It is often applied to data to ensure that each feature has equal weight and to reduce the impact of outliers. Normalization is also scale-invariant, meaning that it does not change the shape of the data as in Figure 2.13. This is important in a framework such as Curve Matching, where the shape of the data is critical to generating accurate similarity scores.

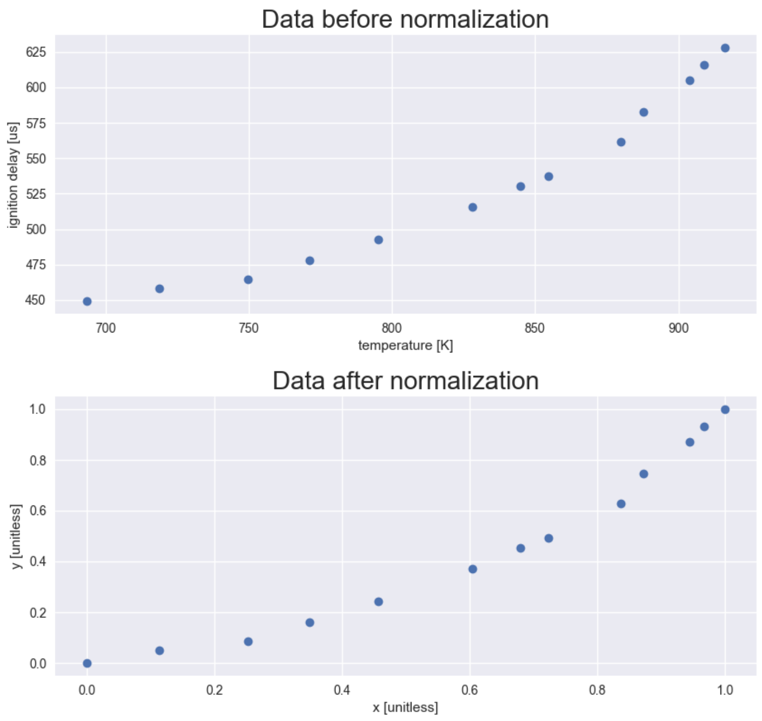


Figure 2.13: Example of min-max Normalization on a set of experimental data.

An example of normalization is min-max normalization, see Equation (2.18), which scales the values of a feature to a range between 0 and 1, where the minimum value is set to 0 and the maximum value is set to 1.

$$x^{'} = \frac{x - min(x)}{max(x) - min(x)} \tag{2.18}$$

This technique is commonly used in machine learning applications and will be used in the development of the methodology in this thesis.

2.5. Fault Injection

fault injection (FI) is a testing technique used to evaluate the resilience and robustness of a system or process in the presence of faults or errors. fault injection can be used in the context of Data Quality to simulate errors or faults in the data and assess how the system or process handles those errors.

In the field of Data Quality, fault injection techniques come in many forms, such as:

- Synthetic data generation: involves generating synthetic data with known errors or faults and using it to test the Data Quality of a system or process.

- Data perturbation: involves deliberately introducing errors or faults into real data and using it to test the Data Quality of a system or process.
- Data masking: involves hiding or removing specific data components to test how well a system or process can handle data that is missing or insufficient.
- Data reordering: This involves changing the order or sequence of data elements to test the resilience of a system or process to changes in data structure.
- Data duplication: This involves duplicating data elements to test the resilience of a system or process to redundant data.

Data Quality analysts can assess the efficiency and dependability of Data Quality management systems and processes using these fault injection techniques, as well as spot any potential weaknesses or vulnerabilities that require attention.

There are several examples of this technique applied to Data Quality in the literature, for example in the paper [43] the authors present a fault injection tool for testing the quality composite services. Basically, data exchanged within various services are perturbed and the consequences of this change on the final result are analyzed, proposing some metrics for an objective evaluation of the deterioration of services.

In [44], on the other hand, a methodology is proposed for emulating failures in data and how this can be used in the study and development of ML applications. This methodology consists of the implementation of a component, called Fault Injector, that simulates faults in the data and from them generates different ML models: after several simulations, it is able to return which model performs better when trained with compromised data and provides analysts with a tool that can understand how to train certain models to be fault tolerant.

Still concerning ML applications, in [3] the authors propose a fault injection approach to evaluate the performance of different algorithms. Unlike [44], this time the technique is applied on data but considering perturbations from the perspective of worsening the Data Quality dimensions considered, in this case Accuracy and Completeness.

The fault injection technique has already been applied to Curve Matching [4], but with a different goal. The previous approach focused on improving the quality of the framework and its specific application through improved data preprocessing. In fact, the application of fault injection has successfully shown some points of improvement for the preprocessing phase: what wants to emerge from this thesis is that even in these phases uncertainties can arise that worsen the quality of the final result, so an analysis of each stage of the pipeline must be made from different points of view.

2.6. Conclusions

In this chapter, the concepts of Model Validation and Data Quality were presented to give the reader an overview concerning the current state of the art in the field.

From the viewpoint of the thesis, it was important to analyze how in the development of predictive models there is no fixed step-by-step procedure for validation: this is because each particular model has different validation and accuracy requirements based on its application domain.

In addition, it was discussed how the science of Uncertainty Quantification plays an important role in the development of such models. Still there is no reference method for the evaluation of all the uncertainty sources implicated in the development process.

As a matter of fact, existing techniques are able to recognize various types of uncertainty (such as aleatory and epistemic uncertainty) in such a way as to help modelers in developing more accurate models: nevertheless, the persistent problem is that these techniques, as well as some validation methods, are not suitable for all types of models.

It must also be considered that the data fed to the models may be of low quality, even without the modeler's knowledge, and that developing models with low quality data automatically brings low-level output, as explained in the GIGO phenomenon, no matter how perfect the model may be.

In addition, it is emphasized in the state of the art that in data pipelines there is always data manipulation and that any evaluation tools take into account Data Quality dimensions as they are very important for the evaluation of the final result.

fault injection techniques have been successfully applied for the evaluation of the performance of different systems, including data processing pipelines. However, today, there are still few technologies that support ad-hoc fault injection, and a full integration of a fault injection technique with uncertainty is still missing. This highlights the importance of developing a comprehensive methodology for evaluating the impact of uncertainty together with bad Data Quality.

Once these concepts are clarified, it is clear how the work proposed in this thesis moves toward a standard methodology for assessing the impact of low-quality data in data-centric model development pipelines: through the measurement of the most widely used Data Quality dimensions in science along with the use of other techniques, a tool will be provided for modelers to objectively assess the reliability of the validation method used.

3 | Methodology

This chapter presents the design choices that guided the research of the methodology chosen for the analysis. The methodology used will be described in depth, including the tools and data used to carry out the analysis, and the context in which this study is carried out will also be explained to clarify the environment in which the experimentation took place.

In Section 3.1 an overview of the research process is given, highlighting the research questions, research objectives, and contributions of this study.

In Section 3.2, the application context of Curve Matching in which the research is situated is explained, and how they developed methodology stands as an analysis tool in pipelines for experimental data.

Then, Section 3.3 contains the core of the methodology, i.e., the type of uncertainty sources used in fault injection, the metrics used for quantifying ambiguity in the Uncertainty Assessment phase, and the explanation of the Uncertainty Propagation phase.

Section 3.4 will explain the design of the Uncertainty Assessment and Propagation results: specifically, the logic behind the results shown (e.g., aggregations, normalizations..), and the metrics used will be explained.

3.1. Research Overview

This thesis aims to investigate how uncertainty propagates in a data processing pipeline, which is used to process and analyze experimental data by building a model capable of simulating the underlying physical process from which it originates.

Specifically, the focus of the thesis is understanding the impact of sources generating uncertainty as they generate ambiguity in the pipeline result. The research carried out in the thesis was done by studying the impact of uncertainty on a validation metric (Curve Matching) and thus the impact it has on model evaluation. To achieve this, the following research questions have been formulated:

1. Is it possible to quantify the impact of sources of uncertainty for each stage of the pipeline?
2. Is it possible to understand how this impact is reflected in the ambiguity of the outcome?
3. Is it possible to provide a tool for analyzing the relationship between sources of uncertainty and ambiguity in results?

To answer these research questions, an approach was developed to measure the ambiguity produced by various sources of uncertainty about the pipeline and individual stage outcomes. Uncertainty will be generated by subjecting the experimental data to the technique of fault injection. The propagation of the impact of uncertainty on the final result will be evaluated by visualization and Spearman's coefficient of Equation (2.16).

3.2. Case Study

The context of this thesis is the field of data processing for building models from experimental data. In many scientific fields and industries, models are important tools for understanding and predicting complex phenomena.

However, building accurate and reliable models can be challenging due to several issues, such as the various sources of ambiguity that affect the final result of data processing. Indeed, in the case of a multi-stage pipeline such as the one analyzed here, various uncertainty-generating factors must be taken into account, some related to the experimental data and others due to its repeated processing.

It is not surprising that the data may be of poor quality their collection is often complicated for various reasons Section 2.1, and consequently the quality of the model also deteriorates (as the GIGO phenomenon in Section 2.3 describes). In this field, research on uncertainty quantification and propagation is extensive, but each solution is linked to a specific approach or conditions, and statistical techniques or approximations are often used, introducing further factors of uncertainty into the analysis.

The focus of this study is to develop a general methodology for analyzing the propagation of uncertainty in a multi-stage data processing pipeline. Specifically, this methodology will assess the impact of different transformations on the quality of data and how this is reflected in ambiguity in the result.

The methodology was designed to investigate the propagation of the impact of different sources of uncertainty through the various stages of the data processing pipeline, from data collection to model validation. This was done by measuring the impact at the state level by Mean Absolute Error and at the pipeline level by the change in the final score.

The development of this methodology is aimed at analyzing possible sources of uncertainty and their impact on pipeline results for experimental data. It is designed to be adaptable to different types of pipelines by analyzing the impact on the ambiguity of the results of each stage regardless of the type of input and output being handled.

The main sources of uncertainty identified, as highlighted in Section 2.2, are the uncertainty inherent in the experimental data, the aleatoric and epistemic uncertainty in the numerical simulations of the experiments, the functional estimation phase, the calculation phase of the dissimilarity indices and, finally, the bootstrapping performed to calculate the final index.

Previous studies on the propagation of uncertainty are based on the use of data where the uncertainty has been quantified in some way, as in [32], or for mathematical simplification of the models, surrogate models are used that introduce approximations into the analysis ([29, 32]): the approach used in this thesis treats uncertainty from a different point of view, i.e. it is injected into the data. As we will see, this is done for every kind of input for a pipeline stage.

3.3. Design

3.3.1. Overview

From the analysis of the context in which the research is carried out and the current state of the art, there is a need for a general approach to the quantification of uncertainty and its propagation in computing applications aimed at the realization of predictive models capable of abstracting complex physical phenomena. The outcome of this study will be a methodology capable of providing a degree of confidence in the validation metrics used on the model, and thus evaluating the model itself, and through this methodology providing guidelines for reproducing the same analysis on a wider range of scientific applications.

This methodology will be developed using the multi-stage Curve Matching pipeline of Section 2.1.3 as a case study, analyzing the impact of various sources of uncertainty on each stage and on the pipeline itself. The sources generating uncertainty will be in the form of percentage perturbations, representing numerical uncertainty in the experimental data, and in the percentage worsening of three Data Quality dimensions chosen for analysis.

The key components of this methodology will be:

- The multi-stage data processing pipeline of the Curve Matching framework, whose stages will be analyzed to detect the presence of uncertainty.
- The Data Quality dimensions were analyzed to understand the impact of transformations on the data and therefore on the final output.
- The transformations performed on the data: in addition to those performed at each stage of the pipeline, the data are perturbed according to certain criteria explained in the following sections, to understand how a deterioration or improvement of the Data Quality affects the final output.
- The metrics used at each stage to assess the impact of uncertainty on the pipeline: the Mean Absolute Error (MAE) and ΔCM.

In practice, for each input at each stage of the pipeline, the Data Quality dimensions are quantified (their measurement is explained later in this chapter) and each dimension is subject to an incremental percentage degradation, and the input is subject to the transformations that occur at each stage of the pipeline. Each input, in addition to the deterioration of the Data Quality dimensions, is subjected to a percentage perturbation corresponding to different levels of numerical uncertainty in experimental data. Each output is compared with the results obtained from the data under perfect conditions (i.e. the original input, without perturbations) and the MAE indicates the ambiguity introduced in the result.

This approach is innovative because, unlike the rest of the research carried out so far, it will use the dimensions of Data Quality (which have been very successful in data research recently) to assess the reliability of a validation tool, in this case, curve matching, regardless of its technological level. This makes it possible to determine whether the quality of the results of a pipeline also depends on the quality of the input received (in line with the GIGO philosophy).

Now follows a roadmap to the next subsections:

- Data Processing Pipeline: this describes the different stages of the data processing pipeline and how uncertainty-generating factors will be assessed for each stage.
- Data Quality Assessment: this subsection will contain the description of the Data Quality dimensions analyzed and how they are quantified.
- Fault injection: this subsection describes the technique used to simulate the impact of different factors on the pipeline. It will be performed on input data at each stage of the pipeline, each time with a different granularity of degradation.

- Uncertainty Assessment: this subsection will provide details on the method used to assess the impact of the fault injection technique on the pipeline. The Mean Absolute Error will be used as a metric to evaluate such impact, along with the variation of the Curve Matching score, namely ΔCM.
- Uncertainty Propagation: in this subsection, the results of the uncertainty assessment will be used to understand the propagation of the uncertainty. Through visualization and Spearman's coefficient, the relationship between uncertainty-generating factors and ambiguity in the output will be assessed.

3.3.2. Data Processing Pipeline

The data processing pipeline is the concretization of the Curve Matching framework of Section 2.1.3, which is the case studied to develop the methodology in this thesis. The pipeline consists of three main stages: functional estimation, computation of dissimilarity indexes, and bootstrapping (Figure 2.2).

The functional estimation stage, which is the first stage of the data processing pipeline, receives two different types of data from SciExpeM ([12]) as input; these data types will be explained in more detail in a later section of the thesis.

The experiment data from the experiment database, which is subject to the innate uncertainty of experiments, is the first input. The second input is the simulation data computed from the model database. These data are susceptible to model-related uncertainties, including *aleatoric* and *epistemic* uncertainties([24]).

As seen in Section 2.1.3, various mathematical operations are used in the functional estimation stage to compute an estimated curve for each of the two inputs. The decision of the knots is the most crucial aspect of this stage in terms of uncertainty issues. The positions of the knots, which are directly influenced by the input data, determine the estimate's general form. Other mathematical operations are also performed at this stage, such as normalizing the input data before computing the knots. The output of this stage is the functional estimation for each input, which is used as input for the next stage of the pipeline. This estimation provides an approximation of the function that describes the relationship between the input and output variables, and it is essential for the subsequent stages of the pipeline.

The quality of the functional estimation is affected by the uncertainties in the input data, and it will be assessed through the use of the MAE metric.

The second stage of the pipeline takes as input the two functional estimates obtained in the first stage: one for the experimental data and the other for the model data. The dissimilarity measures are computed using the norm and scalar product of the two functions. These measures are then normalized and averaged to obtain a final index. The normalization of the dissimilarity measures can create uncertainty due to the loss of generality in this operation.

The third stage of the pipeline is bootstrapping, which is performed on each input curve. The results are then averaged to compute the final index. Bootstrapping is performed to account for uncertainties introduced in the previous stages. However, the averaging process can also introduce uncertainty due to loss of generality, and thus it is also analyzed for uncertainty issues.

It is important to note that these two stages are closely related as the bootstrapping process is easily controlled by a parameter in the Curve Matching framework. As described in the next chapter, this parameter can be tuned to fit the specific needs of the user and its impact on uncertainty can be evaluated.

At each stage of the pipeline, the Data Quality dimensions of Accuracy, Completeness, and Consistency will be assessed. Specifically, for each input at each stage, the Data Quality dimensions will be intentionally worsened multiple times to evaluate the impact of bad Data Quality on the pipeline.

To assess the propagation of uncertainty through the pipeline, the Mean Absolute Error (MAE) metric will be used. A high MAE value indicates high uncertainty, while a low MAE value indicates low uncertainty. By measuring the MAE at each stage of the pipeline, we can evaluate how uncertainty propagates and accumulates through the different stages of the pipeline.

Overall, the data processing pipeline is a key component of the methodology being developed in this thesis, and the assessment of Data Quality and uncertainty will allow us to evaluate the effectiveness and reliability of the methodology.

3.3.3. Data Quality Assessment

Data Quality is a critical aspect of the methodology being developed. To assess the impact of Data Quality on the pipeline, we will consider three dimensions: Accuracy, Completeness, and Consistency.

Accuracy will be measured as the percentage of accurate data points in the entire experiment. For each experiment, we will establish a baseline accuracy level based on expert knowledge of the field and compare it against the accuracy level achieved with the methodology. This will allow us to determine the impact of the methodology on accuracy and assess its effectiveness.

$$Accuracy = 1 - \frac{n}{N} \tag{3.1}$$

where n stands for the number of data points in the experiment perturbed and N stands for the total number of data points in the experiment. As mentioned in Section 2.3, in this methodology we are going to consider for experimental data their semantic accuracy, as it embodies the similarity of the data obtained from the experiment with what can be data of the real phenomenon.

Completeness will be measured as the percentage of data points fed as input, versus the total number of data points available for the experiment. This will allow us to evaluate the completeness of the data at each stage of the pipeline and determine how it affects the performance of the methodology.

$$Completeness = \frac{n}{N} \tag{3.2}$$

Completeness is an easy dimension to measure, in fact in the implementation of the methodology we will see that to evaluate different degrees of completeness it will suffice to subtract a few points from the experiment and as we go along understand how the degradation of this dimension generates uncertainty. The behavior that is expected from this operation is, of course, that as the data are gradually lacking, the output of the relevant stage under analysis will worsen, since it will be increasingly difficult to mimic the original phenomenon.

Consistency will be considered as the percentage of consistent data points in the experiment. Consistency refers to the degree to which the data is free from errors and anomalies.

For each input experiment, we will consider each experimental data as if from different experiments. As explained in Section 2.3, we will create a kind of functional dependence between the experimental measurement under analysis and other dummy experiments.

This dependence will consist of taking the perturbed data and simulating other experiments, making sure that they fall within a range that depends on the injected uncertainty: the less consistent the experiment under analysis, the larger this range will be.

The simulation of experiments will be explained in Section 3.3.4 and will consist of taking the input data as the mean of a statistical distribution, and each experiment will be a sample of this distribution.

What will be taken as the final input will be the mean of these simulated experiments:

clearly, in a consistent experiment this mean will not deviate much from the original point (remember always perturbed, as will be explained in the fault injection process), while for a not very consistent experiment the mean might differ considerably from the original point.

Unlike the other two dimensions that are easier to measure, it is not possible to give a coincident formula that expresses the evaluation of consistency in the methodology.

Instead, we can assess that consistency is directly proportional concerning the parameters α and β of the Beta distribution from which the experiments are sampled. Specifically, as the values of α and β decrease, the resulting samples from the distribution become less consistent and more dispersed, indicating a lower degree of consistency in the experiment.

$$Consistency \propto (\alpha, \beta) \tag{3.3}$$

It is important to note that the three dimensions will be considered experiment-wise, meaning that all data observations referring to one experiment will reflect the same quality but the granularity of fault injection performed on them will be different. For Accuracy and Completeness, the changes will be global, that is, the changes will impact the entire data set of the experiment. As for Consistency, the changes will have a granularity equal to the individual data, as will be clarified in the next section. This will allow us to maintain consistency in the assessment of Data Quality across the pipeline and ensure that the impact of Data Quality is accurately captured.

3.3.4. Fault Injection

As mentioned earlier in the 2.5 section, the application of fault injection techniques has proven to be a successful strategy for asserting the impact of data affected by bad Data Quality on data pipelines, and on systems that process data in general. An example very close to the topic covered in this thesis can be found in the paper [3], where the authors analyze the impact of Data Quality dimensions on a pipeline making use of Machine Learning patterns, using the fault injection technique to worsen, in percentage, the dimensions under analysis.

Instead, the work carried forward in this thesis resumes [4], with the difference that fault injection will be enriched with several injections of uncertainty and the dimensions of Accuracy and Consistency. What will be done instead in the methodology implemented in this paper will be the integrative assessment of worsening not only in Data Quality but also the consideration of uncertainty in it.

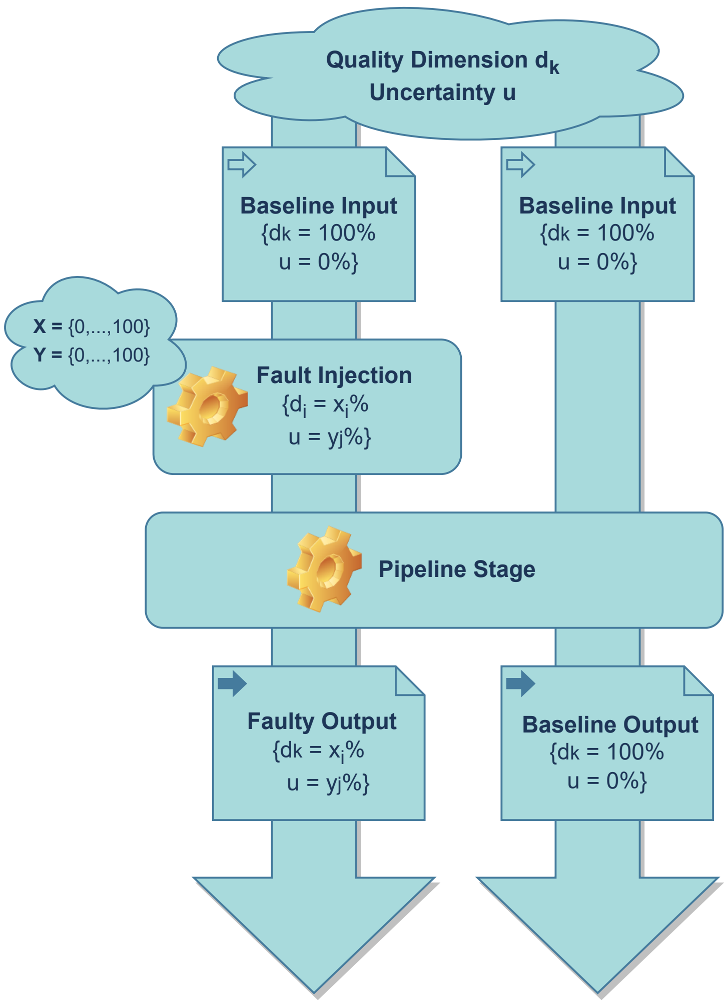


Figure 3.1: Selection of baseline and comparison between faulty and baseline output.

The fault injection process involves the input data being processed in three steps, and then fed to the pipeline stage under analysis (or the entire pipeline), as can be seen in Figure 3.2: the three steps are those of Normalization, Uncertainty Injection, and Data Quality Degradation, which will be explained below.

The fault injection technique involves comparing the output of one stage of the pipeline with a faulty output and one taken as a baseline(Figure 3.1). For each DQ Dimension k, input is taken that corresponds to data of maximum DQ Dimension k and 0% uncertainty. This then undergoes the fault injection process: the faulty output is then given as input to the pipeline stage under analysis. The result of the stage is then compared with the result obtained from the baseline input: this process is repeated for each DQ dimension k, for each degree of quality in and for each degree of uncertainty injected. Ultimately, this allows us to compare the output of the stage with fault-injected inputs against the output of the stage with the maximum quality and 0% uncertainty input, giving insights into how the CM pipeline handles uncertainty and Data Quality issues.

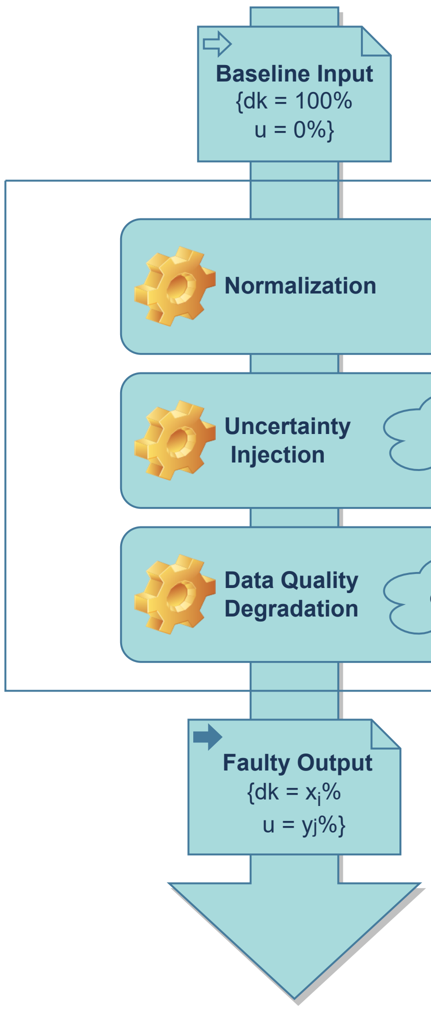


Figure 3.2: Fault injection with three steps: Normalization, Uncertainty Injection, Data Quality Degradation.

As a premise, given the variety of input data sizes from the experiments under consideration, before being subjected to fault injection the data will be normalized: this will help in the final assessment of uncertainty, since through MAE we will have the possibility of quantifying the uncertainty introduced knowing that the input data are scaled. This transformation introduces a loss of generality in the data, however, it does not affect the analysis since what is of interest is the output of the analyzed stage and the uncertainty generated by it.

The normalization technique used is Equation (2.18), which as written earlier in Section 2.4.1 is a scale-invariant transformation. This means that it does not change the shape of the experimental data, which in the case of Curve Matching is precisely the dimension under analysis.

The step of Uncertainty Injection consists of different levels of uncertainty being injected into the data: starting from a level of 0% which will be taken as the basis for the uncertainty calculation that takes place later, we arrive at 100% uncertainty, with steps of 10%. The uncertainty percentage represents the amount that is subtracted or added (randomly) to each input data.

The baseline normalized data, which has 100% quality of dimension d_k and 0% uncertainty, serves as the input for this technique. Suppose we are at the k-th dimension of Data Quality, the x-th degree of uncertainty, and the y-th degree of Data Quality. At this iteration, a random x% perturbation is made concerning the actual value of the data:

$$ND = \{n_0, ..., n_N\}$$
$$U_x = n_i \pm random() * n_i * x, \quad i = [0, ..., N]$$

With ND the baseline normalized data, random() a random number between 0 and 1, and x the percentage of uncertainty injected.

This means that the output of this method will be data of dimension k with x% uncertainty and 100% quality level (this step is before the Data Quality Degradation).

This method helps us understand the pipeline's response to uncertainty and how it affects the pipeline's performance. We can examine how the pipeline responds to different levels of uncertainty and how it affects the results by introducing uncertainty at different levels.

So, Data Quality dimensions are chosen (those in Section 3.3.3) to be considered for fault injection analysis and application: for each dimension chosen, different levels of degradation were chosen.

Taking normalized and perturbed input data, the assumption is made that each dimension is 100% correct as if the data had no errors, then gradually they will be degraded down to 0%. The measurement of dimensions was defined in Section 2.3, while in detail the degradation of each dimension will be explained below. This step will be referred to as the Degradation of Data Quality dimensions.

Accuracy

For each dataset related to each experiment, the percentage of Accuracy represents how many points will be affected by uncertainty and thus suffer perturbation. Thus, considering that we will have fixed the number of inaccurate points in the formula (3.1), we will have an uncertainty of y%.

Considering that B is the baseline input, U_x is the output of the Uncertainty Injection process with uncertainty x and y the degree of accuracy:

$$\begin{aligned} B &= \{b_i, ..., b_N\} \qquad & i &= [0, .., N] \\ U_x &= \{u_j, ..., u_N\} \qquad & j &= choice(i, y * N) \end{aligned}$$

and we will get as output O of the fault injection for the Accuracy dimension:

$$O_{Accuracy}(x, y) = \begin{cases} b_k & k \in i \setminus j \\ u_j \end{cases}$$

It is important to note that using this approach for accuracy degradation is constraining, because as compared to the other dimensions it is directly proportional to the injected uncertainty. this approach was chosen to differentiate the accuracy dimension from the experimental uncertainty, as mentioned in Section 2.3.1.

Completeness

For each dataset about each experiment, the percentage of Completeness represents how many points will be given as input to the stage under analysis, compared to the initial number of points. Considering that we set the number of points n in formula (formula), having a completeness of y

$$U_x = \{u_i, ..., u_N\} \quad i = [0, ..., N]$$

and we will get as output O of the fault injection for the Completeness dimension

$$O_{Completeness} = \{u_j\}\,, \quad j = choice(i, y * N)$$

Consistency

For each data item belonging to the dataset of each experiment, the percentage of Consistency will represent how consistent the simulated experiments are with each other. As explained earlier, for the evaluation of this dimension, samples will be taken from a Beta statistical distribution constructed on the single point that is going to be analyzed.

The Beta distribution is a continuous probability distribution defined by two parameters α

and β on the unit interval [0,1], in our case shifted such that the center is the experimental observation.

To construct the beta-distributed random variable from which to sample the simulated experiments, we consider a generic experiment Z, and random variables X and Y.

$$\begin{aligned} Y &= [y_0, ..., y_N] \\ X_i &\sim Beta(\alpha, \beta) \quad i \in (0, .., N) \\ Z_i &= X_i + y_i - \frac{1}{2} \quad i \in (0, .., N) \end{aligned}$$

The mean of a beta-distributed random variable, in this case, X, is equivalent to

$$E[X_i] = \frac{\alpha}{\alpha + \beta}$$

By the statistical properties of the mean, fixing $\alpha = \beta$:

$$\begin{aligned} E[Y_i] &= E[X_i - \frac{1}{2} + z_i] \\ &= \frac{\alpha}{\alpha + \beta} - \frac{1}{2} + z_i \\ &= \frac{1}{2} - \frac{1}{2} + z_i \\ &= z_i \end{aligned}$$

We thus obtain that the mean of the distribution Y, which is the one we will use for the samples, is the experimental value z_i, so by doing this we are going to ensure that our simulations are at least consistent with the initial experiment. Moreover, the two parameters of the distribution govern its symmetry: having them equal is equivalent to having a symmetrical distribution, centered precisely on z_i.

Starting with 1, increasing the parameters of the distribution, one can shift it from a uniform distribution to a normal-like distribution with an increasingly narrow "bell."

Figure 3.3 figure explains this concept better: as the parameters α and β increase, the distribution becomes less dispersed, thus generating samples that are more consistent with each other (Equation (3.3)).

So, when the distribution is uniform, we can assume that the consistency among the simulated experiments is 0%, while for a distribution with the narrowest bell, the consistency is 100%.

The moving average of all points generated from this distribution, for each data point, will be taken as a reference and substituted for the initial data point. The choice of moving average is motivated by the fact that, since these are samples taken from a statistical distribution, the mean coincides with the value on which the distribution is centered and thus in our case would be equivalent to the initial experimental sample.

The weights of the moving average will be estimated based on the density of the samples: the simulation must take into account a relatively small number of experiments, since by simulating a large number of experiments the density of them will tend to reach the center of the distribution, i.e., the initial experimental sample.

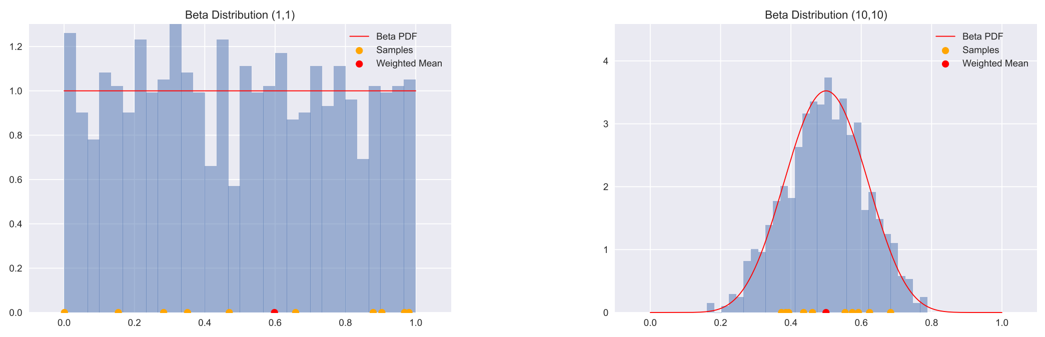


Figure 3.3: Example of sample extraction and calculation of moving average for Consistency dimension evaluation with 10 samples.

Finally, if U_x is the output of the uncertainty injection phase with uncertainty degree x, and W is the set of weighted means of the samples for each $u_i \in U_x$, the output O for the error injection for the consistency dimension is:

$$U_x = \{u_i, ..., u_N\} \quad i = [0, ..., N]$$
$$W = \{w_i, ..., w_N\}$$
$$O_{Consistency} = W$$

Observations

Because these operations can be subject to randomness, in the implementation of this technique these steps will be performed several times to ensure that we have a robust result.

After performing these steps for each dimension, the data is fed to the pipeline stage under analysis, and once the output is produced, it is compared with the results obtained under conditions of 0% uncertainty and each dimension at 100%.

To be able to evaluate the uncertainty for each stage and its propagation, thus what the final uncertainty will be, these operations will be carried out first for each stage and then for the entire pipeline (as if it were a black box), where the input data will be subjected to fault injection and the uncertainty on the final result, in this case, the final Curve Matching index, will be evaluated.

This type of analysis is used to show the robustness of the operations occurring in the pipeline: a likely result would be a worsening of the final index in response to increasingly worse data, however, the analysis could also show anomalous pipeline behavior.

This section presented the overview of the fault injection methodology, highlighting its key steps and rationale. Chapter 4, instead, contains the technical details of the implementation of this methodology, describing each of the algorithms used in more detail.

3.3.5. Uncertainty Assessment

As shown in Figure 3.4, the Uncertainty Assessment is the phase of the methodology that follows the fault injection phase. The result of this stage will be the assessment of the fault injection process, for each of the defined Data Quality dimensions.

What takes place in this phase is the comparison of the output of one of the pipeline stages, simulating its actual behavior in an application, with what would be its output instead in the case that the data is subject to quality problems or uncertainty.

When an experiment is given as input to one stage of the pipeline, it first goes through fault injection and then returns a value that can be considered a kind of prediction of the output under different initial conditions of the same input.

To make the comparison, the Mean Absolute Error is used, which will act as a measure of uncertainty by evaluating precisely the difference between the output obtained under original conditions and the output "predicted" under conditions other than the original conditions.

The final output will be several MAE values equal to the number of degrading iterations within the fault injection process, which in this case will be 10 iterations of Uncertainty Injection, and for each of these 10 Data Quality Degradations, to which another 10 iterations obtained in the original conditions must be added to have a reference output in the original conditions.

The choice of Mean Absolute Error is because each stage output is different, and this was used as a relative metric to measure the impact of fault injection uniformly for each stage analyzed. In addition, to make the impact of each variable analyzed with fault injection more obvious, the MAE obtained from the analyses for each stage is normalized, so that a kind of score is obtained that indicates the importance of each factor on the final error.

With these results, final case analyses can then be made, either by visualization techniques or simple comparison.

The results of the uncertainty assessment consist of the errors produced by each stage of the pipeline when provided with faulty inputs, compared to the results obtained with baseline inputs. By analyzing these errors, we can understand how the pipeline responds to faults and compare the Mean Absolute Error (MAE) to assess how each stage responds to different levels of quality dimensions and uncertainty, both individually and in combination.

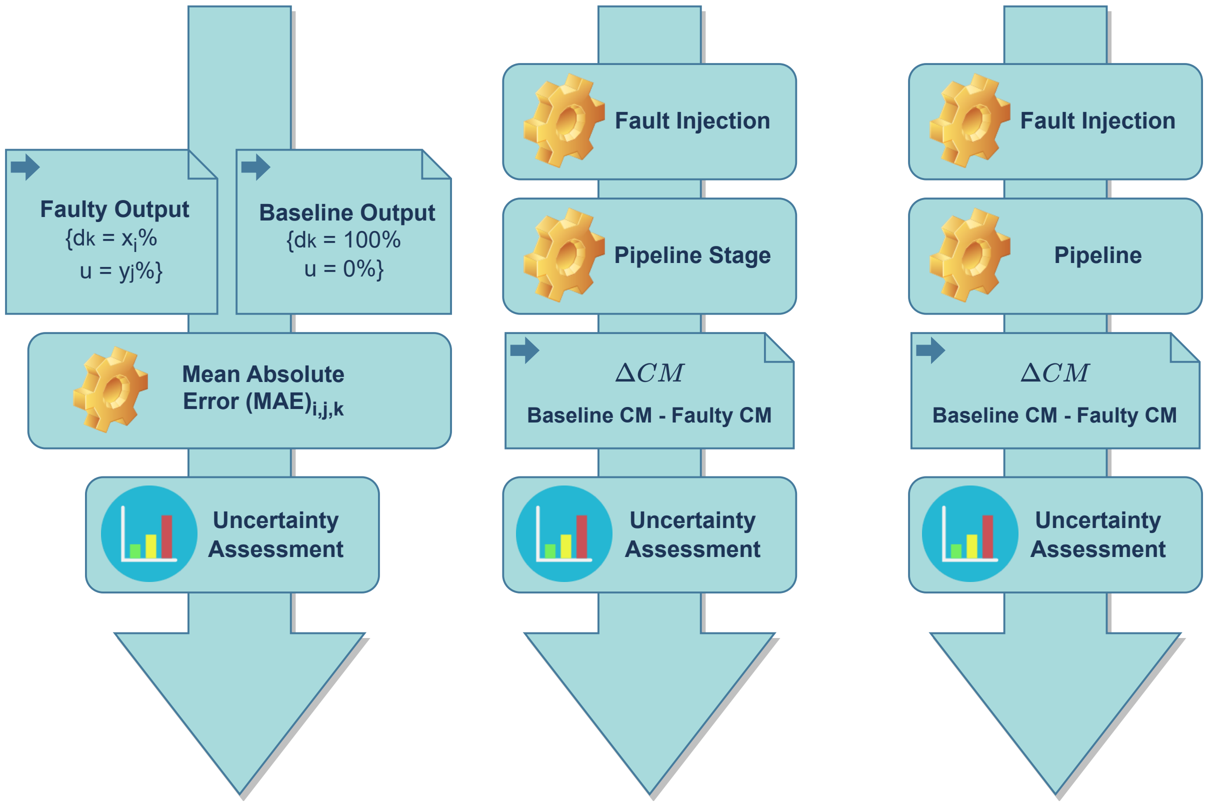


Figure 3.4: Uncertainty Assessment is carried out by comparison and visualization of MAE and ΔCM.

In addition to the MAE, computed with the value of Faulty Stage Output and Baseline Stage Output of Figure 3.4, the Curve Matching score will also be calculated to assess the impact fault injection has on it.

In this way, we will get the impact of each Data Quality Dimension and each degree of uncertainty on the final score as well, so that we get the result of fault injection not only relative to the output of each stage but seeing how the output of each stage propagates within the pipeline to the final result. The Curve Matching relative to the baseline input will have a value of 1, being the comparison of a curve with itself.

fault injection will also be applied on the whole pipeline, to check whether the injected stage-by-stage faults propagate linearly and to check whether the impact on each stage is reflected additively on the final score.

Considerations

MAE (Section 2.1.2) is often used in regression problems where the goal is to predict a continuous value. It is easy to understand and interpret, as it gives a clear idea of how far off the predictions are from the actual values. MAE is also robust to outliers, as it only takes the absolute value of the differences.

In terms of measuring uncertainty, MAE can be used as a way to assess the confidence or uncertainty of a model's predictions. In machine learning, uncertainty can arise due to a variety of reasons such as missing data, noise in the data, or the complexity of the model.

Overall, while MAE is primarily used to measure the accuracy of a model's predictions, it can also be used to assess the level of uncertainty in those predictions by analyzing the variability of the MAE scores or comparing the performance of different models.

In this work, the MAE takes on a dual importance: not only is it used to quantitatively assess the uncertainty in the output, and thus its propagation from the input, but it becomes a measure of the reliability of the various stages Curve Matching framework.

Indeed, one can evaluate the entire pipeline by first entrusting the computation of the score under real conditions, thus simulating its typical use in real applications, then one compares it with scores obtained under different conditions to assess the effectiveness and accuracy of the pipeline.

3.3.6. Uncertainty Propagation

Uncertainty propagation is a critical step in the analysis of data pipelines, as it allows the impact of each fault on the overall ambiguity of the results to be assessed.

Since fault injection provides the evaluation of stages and pipeline outputs concerning different granularities of faults, the results of the uncertainty assessment are plotted to visualize their impact from different perspectives on the outputs. These plots make it possible to visually identify any patterns or trends in the data and to see how the outputs change as the variables are varied.

To quantify the impact of each dimension on the overall uncertainty of the results, we computed the Spearman coefficient Equation (2.16) for each dimension. The Spearman coefficient is a useful tool for identifying which dimensions are most critical for improving the accuracy and reliability of the results and can help guide the design of future data pipelines.

3.4. Design of Results

This section explains how the Uncertainty Assessment and Propagation are designed. Uncertainty Assessment describes the aggregations used in the plots, such as the normalization of the MAE values for each phase. Uncertainty Propagation explains the graphs used to show the correlation between fault injection and measured test results, and the metrics used to measure this correlation.

3.4.1. Results: Uncertainty Assessment

The Uncertainty Assessment section is a critical part of Chapter 5 that aims to provide a quantitative evaluation of the impact of fault injection on the different stages of the pipeline. The first step in this analysis was to apply the fault injection technique to each stage of the pipeline to create artificially induced faults and measure their impact on performance using the Mean Absolute Error (MAE) metric. MAE was chosen as a universal metric to represent the ambiguity of the output under conditions of uncertainty and to make the results adaptable for each output of a stage.

To understand the impact of each injected error, the MAE values for each stage were normalized to bring them to the same scale stage by stage. The next step was to plot the variation of the MAE concerning the level of uncertainty versus the level of Data Quality for each stage and each Data Quality dimension. Since a large number of tests were performed, the median of the results aggregated by the 15 experiments was shown for each plot.

fault injection was then performed again on the stages and also on the pipeline as a whole to compute the variation of the Curve Matching score (ΔCM). The ΔCM is the difference between the perfect Curve Matching score (which is equal to 1), which corresponds to the baseline curve compared to itself, and the CM score computed for the faulted curve. The

ΔCM was calculated for the curves obtained by fault injection of the three stages and then for the whole pipeline.

The results obtained from this section are going to provide a comprehensive understanding of the impact of fault injection on the different stages of the pipeline, and on the pipeline itself, highlighting the importance of considering the uncertainty and Data Quality dimensions in the analysis.

3.4.2. Results: Uncertainty Propagation

The Uncertainty Propagation section aims to explore the relationships between uncertainty, Data Quality dimensions, and the two metrics under analysis: Mean Absolute Error (MAE) and ΔCM.

To achieve this, plots are presented where the x-axis represents the degree of the dimension under analysis (either uncertainty or a DQ dimension) and the y-axis represents the relative metric (MAE or ΔCM). These plots are grouped by all experimental tests for each stage and aggregated by the median.

For each relationship, Spearman's rank correlation coefficient (Spearman's ρ) is calculated. This coefficient assesses whether two variables are monotonically related and, if so, whether the relationship is increasing or decreasing. The coefficient ranges from -1 to 1, with a value of +1 or -1 indicating a perfect Spearman correlation. An increasing monotonic relationship will have a positive sign, while a decreasing relationship will have a negative sign.

By computing Spearman's ρ, it is possible to:

- Understand the impact of each error injected at the input of each stage on the output ambiguity measured by MAE.
- Understand whether this effect on output ambiguity is reflected in the final ambiguity measured by ΔCM.
- Understand the impact of each error injected at the input of the pipeline on the ambiguity of the final score measured by ΔCM.

In summary, the Uncertainty Propagation section provides a deeper understanding of the impact of uncertainty and DQ dimensions on the metrics under analysis, as well as the correlations between fault injection and the resulting output ambiguity for each stage and the entire pipeline.

4 | Implementation

This chapter explains the implementation of the methodology developed in the thesis. It describes the actual translation of the design of Chapter 3 in algorithmic form, explaining the implementation of the fault injection technology with the various Data Quality dimensions analyzed and uncertainty. It also discusses the technologies used and describes the experimental data used to test the methodology.

Section 4.1 presents the technologies used to implement the methodology.

Section 4.2 presents the experimental data used to test the developed methodology and to evaluate the uncertainty generated at each stage and its propagation to the final result.

Section 4.3 describes the implementation of the Chapter 3 architecture integrated with the technologies used.

Section 4.4 describes the implementation of the different steps of the methodology by showing pseudocode of the algorithms used.

4.1. Technologies

The architecture of the methodology presented in the Chapter 3 was implemented in Python, making use of several libraries:

- SciExpeM-API: a library used to retrieve experimental data from SciExpeM[12] database and to calculate the Curve Matching index.
- SplinePoliMi: a library used to calculate the splines of the experimental data and the defective data, to simulate the Functional Estimation stage.
- NumPy[45]: a library used for data management and data transformation.
- Pandas[46]: a library used to use efficient structures for large amounts of data and to analyze them.
- Matplotlib[47], Seaborn [48], Plotly [49]: very common libraries in the field of data analysis, used to evaluate the results of analysis from different perspectives.

The various steps of the fault injection process were implemented by manipulating with Numpy the experimental data collected through the SciExpeM API. The results of the tests performed were fed into various Pandas DataFrames and aggregated for analysis visualization using the aforementioned plotting libraries.

4.2. Data Collection

This section will introduce SciExpeM, a framework for the collection of services for data and model management and analysis[12].

In Section 4.2.1, the why of selecting these data and how they are retrieved will be explored, meanwhile in Section 4.2.2 we will instead see the experimental data retrieved by SciExpeM, which will then be used to test the developed methodology.

4.2.1. SciExpeM

SciExpeM [12] is a widely used data management framework in chemical engineering. It provides a rich set of services to automatically collect, manage, and analyze experimental data and models. With a microservices-based architecture and a data analysis library, the framework allows for the efficient deposition of experiments and dynamic analysis processes.

In the context of this thesis, the experimental data is retrieved from SciExpeM, as Curve Matching is used in the data analysis phase to compare experimental data and models generated on them. While point-to-point validation metrics are not suitable for comparing experimental data and models, Curve Matching provides a curve trend approach that can directly evaluate the model against the experimental data.

The Python library, SciExpeM-API, provided by SciExpeM was used in this thesis to retrieve the experimental data. The library allows filtering the database based on certain criteria such as category or experiment ID. The experiments were first viewed directly from the SciExpeM platform and then filtered through the API to obtain the experimental data needed to test the approach developed in this thesis.

The experimental data from SciExpeM are then used to apply fault injection to the Curve Matching pipeline, simulating any errors in the data in a real application and seeing how the pipeline reacts.

Model data were not used because, as emphasized again, the approach simulates uncertain and bad quality data: to calculate any error, and thus the introduction of uncertainty into the various stages of the pipeline, it is necessary to compare the output of each stage with a faulty input with the output of the stage with a baseline input.

4.2.2. Experimental Data

In this section, the experimental data obtained through the SciExpeM API will be presented and analyzed. The data retrieval process was carried out using the SciExpeM-API Python library, which provides access to the data management framework of SciExpeM. The experimental data were filtered according to specific criteria to obtain a representative dataset for testing the approach proposed in this thesis.

The criteria for selecting the experimental data were based on three factors, visually evaluated on the SciExpeM framework:

- the category of the experiment
- the number of data points available
- the shape of the functional estimation computed through the SplinePolimi library

The category of the experiment was chosen to have more heterogeneous data. Five categories were chosen:

- ignition delay measurement (Figure 4.1)
- outlet concentration measurement (Figure 4.2)
- concentration-time profile measurement (Figure 4.3)
- laminar burning velocity measurement (Figure 4.4)
- jet stirred reactor measurement (Figure 4.5)

The number of data points was taken into consideration as an indicator of the level of detail provided by the experimental data. Finally, the shape of the functional estimation was used to ensure that the experimental data presented a variety of patterns and behaviors.

In particular, the functional estimations were categorized as linear or wiggly based on their shapes. Linear estimations were identified as those computed with a low number of points, resulting in a relatively smooth curve.

On the other hand, wiggly estimations were characterized by a high degree of variability in the vertical direction, indicating a complex and non-linear behavior of the system under analysis. The selection of these shapes was motivated by their high frequency in the experimental data available on SciExpeM, as well as their potential impact on the performance of the Curve Matching pipeline.

Overall, the criteria used for selecting the experimental data aimed at providing a representative and diverse dataset that could effectively test the proposed approach. The following chapter will present the results of the analysis carried out on these datasets, highlighting the strengths and weaknesses of the Curve Matching pipeline in handling experimental data with different characteristics.

Below are plots of the selected experimental data and functional estimates using splines.

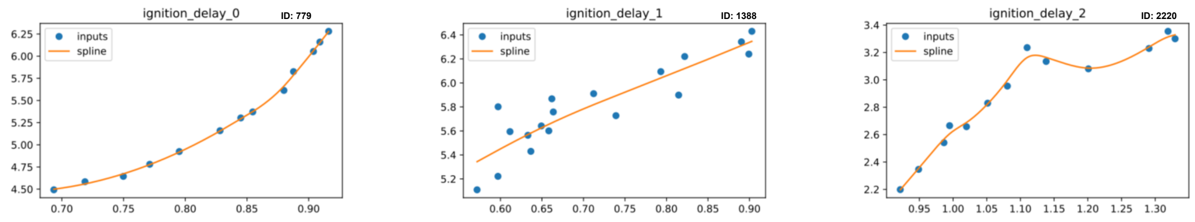


Figure 4.1: Ignition delay measurement experiments. x = [1000/K], y = μs

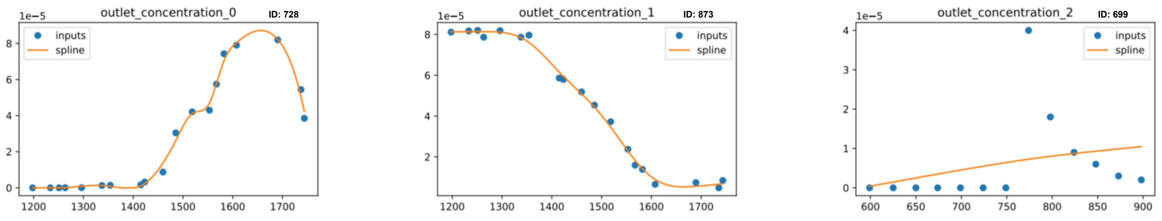


Figure 4.2: Outlet concentration measurement experiments. x = [K], y = [mole fraction]

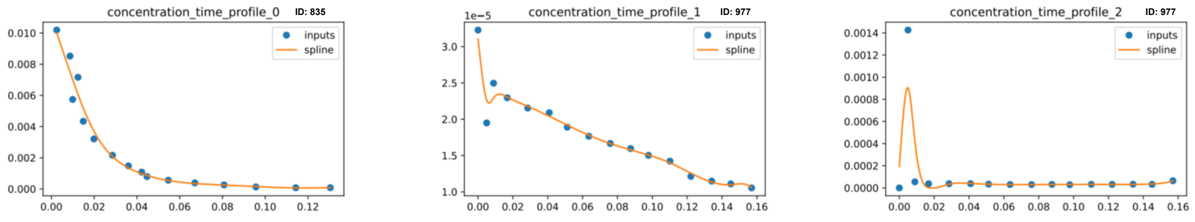


Figure 4.3: Concentration time profile measurement experiments. x = [s], y = [mole fraction]

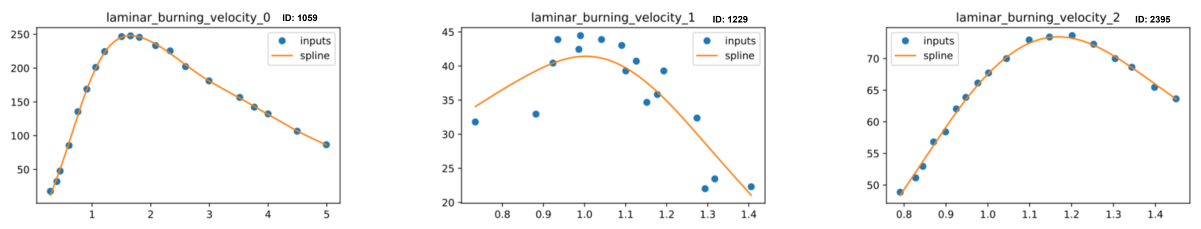


Figure 4.4: Laminar burning velocity measurement experiments. x = [unitless], y = [cm/s]

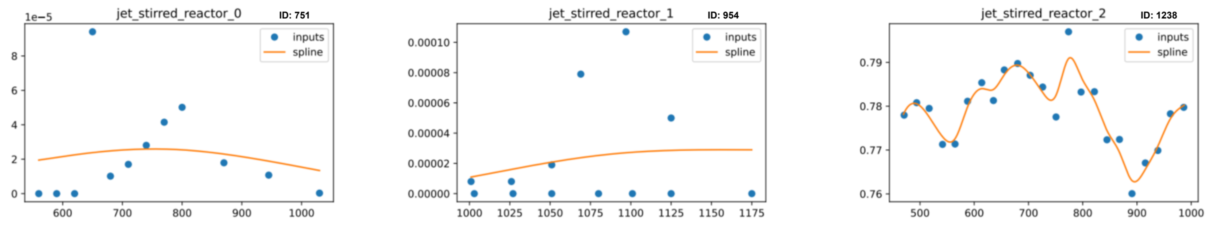


Figure 4.5: Jet stirred reactor measurement experiments. x = [K], y = [mole fraction]

The choice of a small number of experimental data on which to test the developed approach stems from time limitations. However, the choice of data in this type of analysis is important, as pointed out by the authors of [6]. Therefore, the data collected by SciExpeM were chosen by making sure that they were representative of most of the experiments available in the database.

The heterogeneity of the experimental data chosen for this study was a deliberate choice to ensure that the analysis included as many baseline inputs as possible and to reduce the variability of the results. This approach was designed to assess whether any of the experimental data responded particularly well or poorly to fault injection.

By selecting experimental data with different categories, shapes, and numbers of points, we were able to ensure that our analysis was not biased toward any particular type of experimental data. Furthermore, this approach allowed us to test the robustness of the approach and identify any limitations that might arise when applying it to different types of experimental data.

One of the main benefits of using a heterogeneous set of experimental data is that it allows us to better understand the behavior of the pipeline under different initial conditions.

This approach allowed us to gain a deeper understanding of the relationship between the experimental data and the pipeline, and to identify potential areas where the approach could be improved. The heterogeneity of the experimental data selected for this study is a key factor in ensuring that our analysis is robust and provides valuable insights into the behavior of the pipeline under different conditions.

4.3. Architecture

In this section, I will provide an overview of the architecture used in the implementation of the methodology proposed in this thesis. Then I will discuss the architecture of the fault injection technique, which is an important component of the methodology. This section will provide readers with a high-level understanding of the different components and how they work together to support the overall methodology.

4.3.1. General Overview

The methodology focuses on the fault injection technique: specifically, after generating an output with a baseline input and an input degraded by the technique, they are compared by MAE.

This procedure is also followed for CM scores: specifically, fault injection is performed for each stage S in the pipeline, and then the CM score after each iteration of fault injection is compared to that obtained with the baseline input (intuitively, $CM_{baseline} = 1$, since it is the score of a curve compared to itself).

In addition to S stages, the ΔCM is also calculated after fault injection on the pipeline itself.

The Figure 4.6 shows the general implementation after choosing a pipeline stage S, experimental data as baseline input B, a Data Quality dimension to analyze q_i, a quality degree d_j and uncertainty degree u_k.

Thus, after performing all fault injection interactions, all errors for each input combination are obtained and the uncertainty generated at the specific stage can be evaluated for each experiment, Data Quality dimension and uncertainty analyzed.

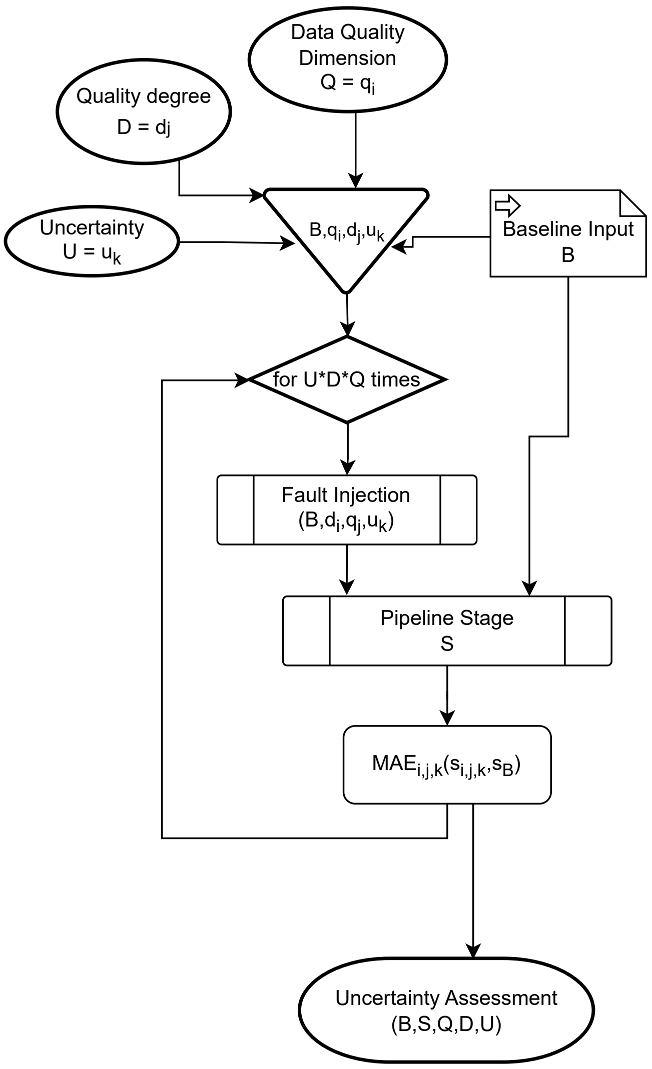

Data Quality Dimension $Q = q_i$
Quality degree $D = dj$
Uncertainty $U = u_k$
B,q_i,d_j,u_k
Baseline Input B
for U*D*Q times
Fault Injection (B,d_i,q_j,u_k)
Pipeline Stage S
$MAE_{i,j,k}(s_{i,j,k},s_B)$
Uncertainty Assessment (B,S,Q,D,U)

Figure 4.6: General architecture of the implementation.

As can be seen in the figure, each baseline input B is subjected to error injection a total of U*Q*D times.

In the proposed methodology, there are 3 dimensions analyzed, while U and D contain 11 elements corresponding to the degrees of uncertainty and quality as they are evaluated, starting from 0% to 100% with steps of 10%.

Therefore, for each of these outputs, the MAE is calculated compared to the output obtained from the stage S with the baseline input B under normal conditions. At the end of the error injection process, we will obtain 3 11x11 matrices corresponding to the MAE values calculated for each dimension in Q according to different uncertainty levels U and quality D.

This process is repeated for each stage of the pipeline to be analyzed, in this case Curve

Matching has 3 stages, and then it is repeated for the stages and the pipeline itself to compute the ΔCM as in Figure 3.4.

Considering also the number of experiments analyzed, i.e. 15 (see Section 4.2.2), and 3 rounds for the MAE plus 4 rounds for the CM, a total of U*Q*D*7*15 = 38.115 tests were performed: given the use of random elements in the implementation of the methodology, these were repeated 10 times and then aggregated to reduce the test uncertainty.

The results obtained are then analyzed graphically from different perspectives; in fact, this high number of tests allows us to analyze the results from different granularities.

The Uncertainty Propagation due to input errors is evaluated by visualizing the Uncertainty Assessment results and calculating the Spearman coefficient to understand the relationship between input errors and ambiguities in MAE and CM results.

4.3.2. Architecture of Fault Injection

In this section, I will present the implementation of fault injection. After presenting the general architecture of the implementation, this is a more detailed look at the different components and their functionalities and how they interact with each other.

fault injection (Figure 4.7) receives as input experimental data B serving as baseline input, a Data Quality dimension q_i, a quality degree d_j, and an uncertainty u_k.

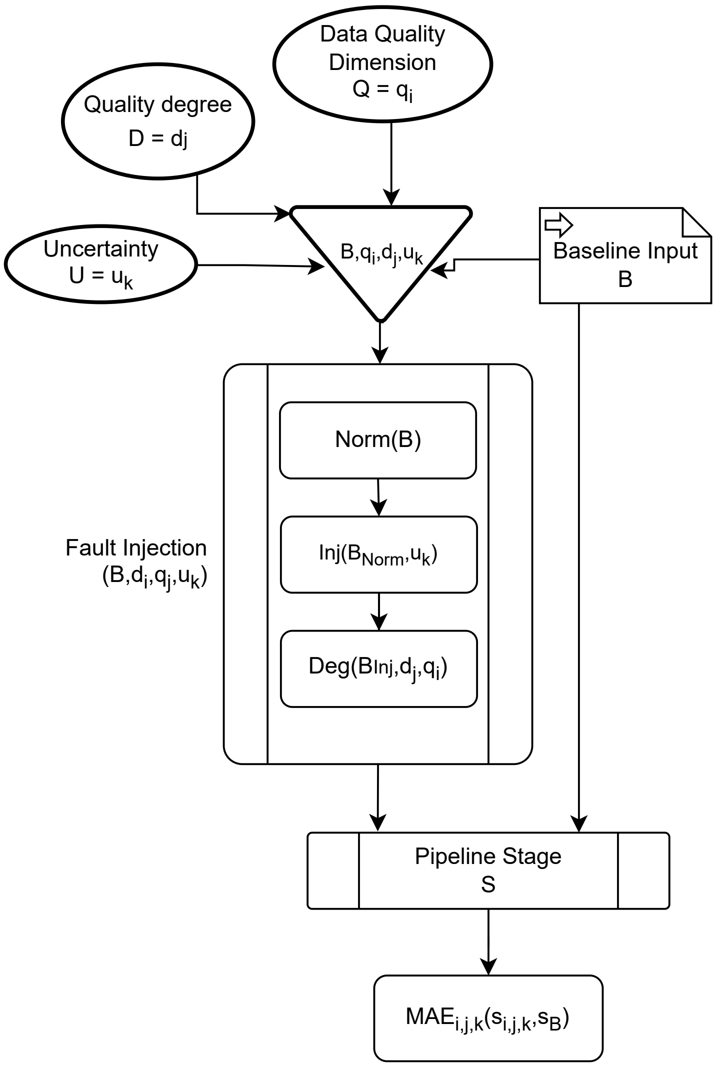


Figure 4.7: Overview of the fault injection implementation.

The first function used is **Norm**, which takes the baseline B as input: it simply calculates the min-max normalization of Equation (2.18) on the received input. As mentioned in Section 2.4.1, this operation scales all data to [0,1] and leaves the shape of the data unchanged. It is used to calculate the MAE on a common scale, otherwise its values would be distorted according to the scale of the input data, which, as seen in Section 4.2.2, have different sizes and values.

The second function **Inj** takes as input the result of *Norm*, B_{Norm}, and returns a data set perturbed by u_k%. For each iteration k of the process, this will be of increasing severity.

The next function, **Deg**, takes as input B_{Inj}, the data produced by *Inj*, and returns a variant of the data with a quality equal to d_j for the dimension under analysis q_i. For each of the dimensions under analysis, we will see in the next section a different implementation of *Deg*, according to what has been written about Section 3.3.4. This operation will therefore be performed for the dimensions Accuracy, Completeness and Consistency, which will be progressively degraded, and the output will be fed to the stage of the pipeline under analysis, S.

To mitigate the effects of randomness on the analysis, each input was subjected to 10 repetitions of the *Inj* and *Inj* functions. This was necessary because the *Inj* and *Deg* functions use different random elements in their implementation, which can introduce variability into the results. Running the functions 10 times attempted to reduce this variability and provide more reliable results.

For each of the inputs, fault injection thus produces a result that is fed into the pipeline stage, which produces $s_{i,j,k}$ as its output. Its result is compared via MAE with the pipeline stage output s_b, which is obtained from the baseline input with the highest quality initial conditions. Each of the i,j,k iterations of fault injection thus corresponds to a value of $MAE_{i,j,k}$.

As explained earlier in Section 3.3.5 and Figure 3.4, for each fault injection the CM score will also be computed, which will be compared to that obtained from the input baseline conditions, and thus a $\Delta CM_{i,j,k}$ for each iteration will be obtained.

The implementation of the *Norm*, *Inj*, and *Deg* functions will be better explained in the following section using pseudocode, in order to reach the maximum level of detail in the explanation of the methodology.

4.4. Pseudocode

Algorithm 4.1 shows the pseudocode of the implemented fault injection technique. It takes as input the baseline input B, the stage S (or the pipeline, which for convenience I will always call S, for CM computation), the Data Quality Dimension q_i, the Quality Degree d_j, and the Uncertainty Degree u_k. As mentioned before, this code is executed n = 10 times to minimize the impact of the randomness of the functions used within the code.

After the fault injection is performed, MAE_n and ΔCM_n are computed and averaged, resulting in $MAE_{S,i,j,k}$ and $\Delta CM_{S,i,j,k}$.

The ΔCM is calculated by subtracting 1, which is a CM corresponding to the unmodified baseline input, and the CM obtained by comparing the curve obtained from iteration i,j,k for stage S with the baseline curve.

Algorithm 4.1 Fault Injection algorithm

```
1: procedure FAULT_INJECTION($S, B, q_i, d_j, u_k$)
2:     $R_1, R_2 \leftarrow [\,]$
3:     $B_{Norm} \leftarrow Norm(B)$
4:     $s_b \leftarrow S(B)$
5:     for $n = 1, ..., 10$ do
6:         $B_{Inj} \leftarrow Inj(B_{Norm}, u_k)$
7:         $B_{Deg} \leftarrow Deg(B_{Inj}, q_i, d_j)$
8:         $s_{i,j,k} \leftarrow S(B_{Deg})$
9:         $R_1[n] \leftarrow MAE(s_{i,j,k}, s_B)$
10:        $R_2[n] \leftarrow \Delta CM_{i,j,k}$
11:    end for
12:    $MAE_{i,j,k} \leftarrow average(R_1)$
13:    $\Delta CM_{i,j,k} \leftarrow average(R_2)$
14:    return $MAE_{i,j,k}, \Delta CMi, j, k$
15: end procedure
```

As can be seen from Algorithm 4.1, the functions *Norm*, *Inj*, and *Deg* are the ones that perform the Normalization, Uncertainty Injection, and Data Quality Degradation operations of Section 3.3.4. *Norm* performs the min-max normalization of Equation (2.18), while now the pseudocodes of *Inj* and *Deg* will be introduced, respectively.

Algorithm 4.2 shows the pseudocode of the function *Inj*, which takes as input B_{Norm}, that is, the baseline input that has undergone normalization by *Norm*, and the degree of uncertainty corresponding to iteration k, u_k.

Algorithm 4.2 Uncertainty Injection algorithm

```
1: procedure INJ($B_{Norm}, u_k$)
2:     $B_{Inj} \leftarrow [\,]$
3:     for $b_i \in B_{Norm}$ do
4:         $sign \leftarrow random([-1, +1])$
5:         $B_{Inj}[i] \leftarrow b_i + sign \times b_i \times u_k$
6:     end for
7:     return $B_Inj$
8: end procedure
```

The degree u_k is a value that corresponds to the percentage of uncertainty injected into the values of B_{Norm}, and ranges from 0 to 1 with steps of 0.1 (corresponding to increasing uncertainty from 0% to 100%).

For each element of B_{Norm}, therefore, the sign of the perturbation to be performed is first chosen with the function *random*, which returns a random element between -1 and +1, and then for each element $b_i \in B_{Norm}$ a perturbation of $b_i \times u_k$ is performed.

Algorithm 4.3 shows the pseudocode of the function *Deg*: This function takes as an input B_{Inj}, i.e., the result of the Uncertainty Injection process, and then according to the Data Quality dimension under analysis, q_i, performs a degradation of the data under analysis so that it has a quality of d_j.

Algorithm 4.3 Data Quality Degradation algorithm

```
1: procedure DEG($B_{Inj}, q_i, d_j$)
2:     case $q_i ==' Accuracy'$ :
3:     $B_{Deg} \leftarrow B$
4:     $indexes \leftarrow choice([0, B.len - 1], size = d_j \times B.len)$
5:     $B_{Deg}[indexes] = B_{Inj}[Indexes]$
6:     return $B_{Deg}$
7:     case $q_i ==' Completeness'$ :
8:     $B_{Deg} \leftarrow B_{Inj}$
9:     $indexes \leftarrow choice([0, B.len - 1], size = d_j \times (B.len - 2))$
10:    $del\ B_{Deg}[indexes]$
11:    return $B_{Deg}$
12:    case $q_i ==' Consistency'$ :
13:    $B_{Deg} \leftarrow [\,]$
14:    for $b_i \in B_{Inj}$ do
15:        $samples \leftarrow Beta(\alpha, \beta, size = 10) - b_i$
16:        $B_{Deg}[i] \leftarrow weighted\_average(samples)$
17:    end for
18: end procedure
```

In the case $q_i == Accuracy$, which we evaluate as the number of perturbed points of B, exactly the indices of the points to be perturbed are taken and inserted into the variable *indexes*.

This variable is filled by the random method *choice*, which selects in the range equal to $[0, B.len - 1]$ several indices equal to $d_j \times B.len - 1$, where $B.len$ is the length of the input baseline B.

Recall that d_j corresponds to the degree of quality that the output of the method will have, so if, for example, in the input $B.len == 10$ and $d_j = 0.5$, the number of perturbed points in the output will be 5, corresponding to an accuracy of 50%.

Thus, the perturbed points of B_{Inj} are assigned to B_{Deg}, which is initialized to B_{Norm}: the function essentially returns the normalized points of B if the uncertainty were injected only in d_j% of the points.

In the case $q_i == Completeness$, again via index selection logic in the variable *indexes*, several points equal to $d_j \times (B.len - 2)$ in the range $[0, B.len]$ will be selected. Since completeness of 0% would correspond to having an empty input, B.len - 2 is chosen as the upper limit of points to remove, since 2 is the minimum number of points for proper operation of all stages and the pipeline.

In case $q_i == Consistency$, for each point $b_i \in B_{Inj}$, 10 samples are taken from a distribution *Beta*, centered on b_i, with $\alpha = \beta$ as parameters of the distribution. Then, $B_{Deg}[i]$ is assigned the weighted average of *samples*, obtained via *weighted_ average*.

Algorithm 4.1 is applied to each experiment in Section 4.2.2 and for each of them the following results are obtained:

- Value of MAE for each level of uncertainty, for each level and dimension of Data Quality, for Functional Estimation, Dissimilarity Measures, and Bootstrapping stages.
- The values of ΔCM, correspond to the change in the Curve Matching score for each iteration of fault injection for the three stages and, in this case, also for the application of the technique to the pipeline.

4.5. Conclusions

This chapter presented a generic implementation of the methodology developed in this thesis to demonstrate the process of obtaining data for uncertainty analysis in the pipeline. The methodology was tested on the Curve Matching pipeline and focused on the analysis of the stages of a pipeline for experimental data.

The goal with which the methodology was implemented was to be able to understand what the ambiguity in the results of a pipeline comes from and to which variable it can be attributed.

A preliminary analysis of the output of the stages using Mean Absolute Error made it possible to understand first of all which of the uncertainty and Data Quality dimensions affect the output the most, and then by analyzing the impact these have on the output of the pipeline (in this case, the Curve Matching score).

In Chapter 5, the results obtained from the fault injection analysis were analyzed both graphically, by plotting them, and analytically through the Spearman rank to understand the relationship between faults and ambiguities in the pipeline.

5 | Results

The objective of this chapter is to show the results of the methodology developed on the Curve Matching experimental data pipeline. In Section 5.1, Uncertainty Assessment results will be presented, i.e., the measurement of MAE and ΔCM for each iteration of fault injection, for each experiment and stage, while in Section 5.2, the relationships between the injected faults and the results obtained in the previous section will be presented. Finally, in Section 5.3 we will discuss the conclusions related to what was seen in the results and then the analysis of the most sensitive stages and variables that most affected the ambiguity of the results obtained in Section 5.1.

5.1. Uncertainty Assessment

This section presents the results of the Uncertainty Assessment. The plots in this section show the relationship between the level of uncertainty and the level of Data Quality for each stage of the pipeline. The metrics representing output ambiguities are plotted on the y-axis, while the degree of quality is plotted on the x-axis. The color of the lines represents the level of uncertainty.

To better visualize the data, Seaborn's [48] relplots were used, which are scatter plots with regression lines superimposed. These plots show the median value of the metric for each stage, aggregated across all experiments. By looking at the plots, we can see the relationship between the degree of uncertainty and the degree of Data Quality on the output metrics. In fact, the x-axis shows the degree of quality with which the test was performed, while the color of the line represents the degree of uncertainty injected through uncertainty injection (see Section 4.4). In addition, for each line, lighter colored bounds are plotted, corresponding to the 95% bounds of the test values.

It is important to note that the degree of uncertainty and Data Quality varies for each stage. Therefore, in order to compare the results, the MAE scores for each stage were normalized so that they could be compared on the same scale for each stage. The plots with MAE on the y-axis show the median of the normalized scores for each stage.

Another important factor to be taken into account when presenting the results of this section concerns the plots of the accuracy dimension. As mentioned in Section 3.3.4 and Section 2.3.1, the tests carried out on the accuracy dimension will have a more linear behaviour, since it also depends on the uncertainty injection Section 4.4. In fact, in these plots, the quality level refers to the number of points disturbed by an amount equal to the uncertainty degree.

Overall, these plots provide a comprehensive understanding of the impact of the injected defects on the performance of each stage. By analyzing the relationship between the level of uncertainty and the Data Quality, we can better understand the impact of each injected fault on the output of each stage.

5.1.1. Functional Estimation

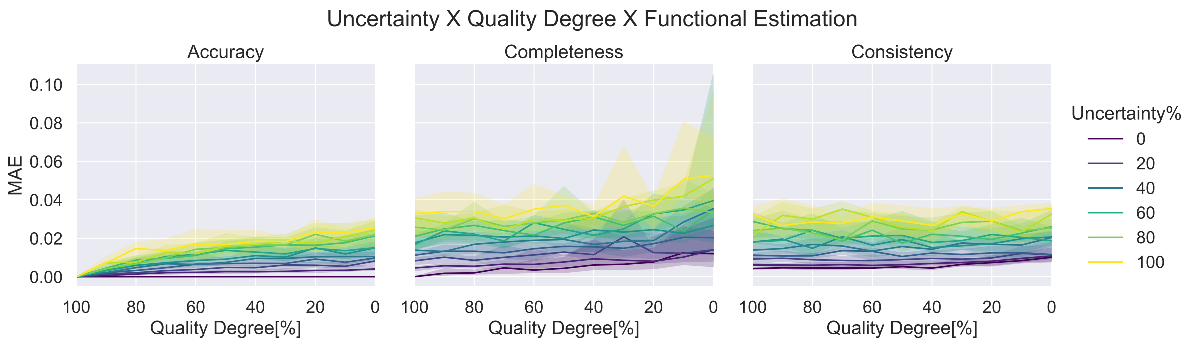


Figure 5.1: Uncertainty Assessment for Functional Estimation: MAE

Figure 5.1 shows the Uncertainty Assessment on the Functional Estimation stage, with the measurement of output ambiguity through MAE.

The increasing trend for both uncertainty and quality for accuracy and completeness, while for consistency the MAE increases only with increasing uncertainty, highlights the importance of both factors in determining the ambiguity of the output. In particular, accuracy and completeness are affected by both uncertainty and quality, while consistency is mainly affected by uncertainty alone.

The plot on the right in Figure 5.1 shows how, from a quality degree of 100% down to 0%, the slope of the MAE remains unchanged, in contrast to the other two plots. Again from the same plot, a more pronounced uncertainty corresponds to an increasingly higher MAE. For an uncertainty value of 0%, corresponding to a purple line, the MAE remains around 0.01. Moving to a more pronounced uncertainty, corresponding to increasingly clear lines leads to an uncertainty of around 0.04. This suggests that improving Data Quality and reducing uncertainty can both have a positive impact on the accuracy and completeness of the functional estimate, but improving consistency may require additional measures to reduce uncertainty.

The increasing impact of uncertainty for each level of uncertainty in all three DQ dimensions suggests that uncertainty has a compounding effect on output ambiguity. As uncertainty increases, so does the impact on the MAE, regardless of the specific DQ dimension considered.

Completeness has the largest impact on the MAE, probably because it measures how well the functional estimate captures all aspects of the data. In fact, for an increasingly worse degree of completeness, the MAE value goes as low as 0.1, as opposed to a maximum of about 0.04 for consistency and 0.03 for accuracy. Decreasing Data Quality results in a steeper slope for completeness, indicating that even small changes in quality can have a significant impact on performance. Accuracy also shows this trend, but since it is more related to uncertainty, it is possible that its impact is less related to the quality degree.

The large outliers in completeness suggest that there are specific instances where Data Quality has a particularly large impact on output. These spikes could be due to specific outliers in the input data or due to limitations in the functional estimation method. Understanding and addressing these outliers can be important for improving the overall quality of the output.

The overlap of the boundaries of different levels of uncertainty and quality in low quality and high uncertainty conditions suggests that it may be difficult to distinguish the contributions of these factors in determining the ambiguity of the output. This reinforces the need to both improve Data Quality and reduce uncertainty as much as possible, especially in situations where both factors are high.

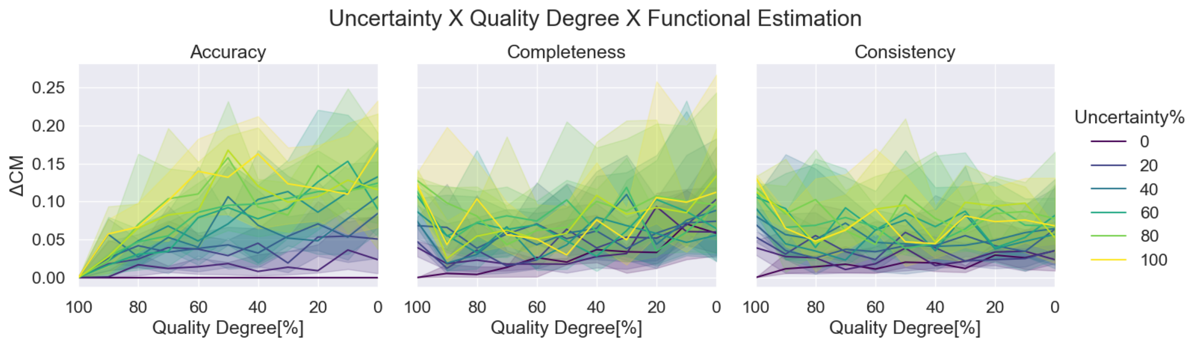


Figure 5.2: Uncertainty Assessment for Functional Estimation: ΔCM

The results of Figure 5.2, which represents Uncertainty Assessment measuring ΔCM on Functional Estimation, show an increasing trend for both accuracy and completeness, similar to what was observed in the MAE analysis. In addition, the effect of uncertainty on ΔCM is consistent with what was observed in the MAE analysis, with a clear increasing trend for each level of uncertainty in all three DQ dimensions.

However, the effect of DQ on the ΔCM is not as straightforward. While accuracy shows a clear increasing trend with increasing DQ, completeness and consistency do not show consistent increasing trends in ΔCM. In fact, at some points the median ΔCM decreases with decreasing DQ degree in these dimensions.

In fact, for completeness in Figure 5.2 you can see from the yellow line, representing the maximum uncertainty, decreasing the quality from 100% to 90% results in an MAE of about 0.1 less. The same thing can be seen in the consistency plot in a noticeable extent when going from 100% to %80 per cent.

This could be due to the fact that completeness has a greater impact on the functional estimate, but all the evaluated errors could change the shape and trend of the evaluated curves, resulting in the observed variations in ΔCM.

The plots also show large outliers for all tests, which is consistent with the observation made in the MAE analysis. This suggests that the impact of completeness on the functional estimate is significant and can result in large variations in the ΔCM values.

Finally, the boundaries of different uncertainty levels and DQ grades overlap under low quality and high uncertainty conditions, suggesting that the contribution of uncertainty and Data Quality is indistinguishable. This finding underscores the importance of maintaining high Data Quality and minimizing uncertainty to ensure accurate functional estimates.

5.1.2. Dissimilarity Measures

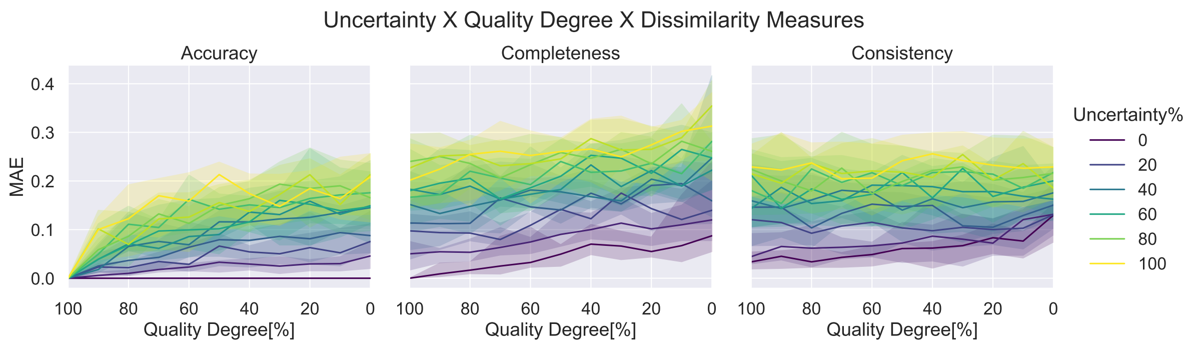


Figure 5.3: Uncertainty Assessment for Dissimilarity Measures: MAE

Figure 5.3 shows the Uncertainty Assessment on the Dissimilarity Measures stage, with MAE as a measure of output ambiguity.

Both accuracy and completeness show an increasing trend with increasing uncertainty and decreasing quality and contrary to what was seen for the Functional Estimation stage, consistency has a similar impact on the MAE compared to accuracy and completeness. Especially for a low uncertainty, corresponding to the darker lines in Figure 5.3, going from a quality degree of 100% to 0% shows an increasing MAE up to 0.1.

The magnitude of the impact of uncertainty is greater for dissimilarity measures than for Functional Estimation, suggesting that uncertainty has a greater impact on this stage. The maximum MAE value for Figure 5.1 is in fact 0.1 while Figure 5.3 shows an MAE for the completeness dimension of up to 0.4. There are fewer major outliers compared to Functional Estimation, with only a few spikes around 0% completeness.

The boundaries of the different uncertainty and quality levels overlap in low quality and high uncertainty conditions, suggesting that the contribution of uncertainty and Data Quality is indistinguishable in these cases.

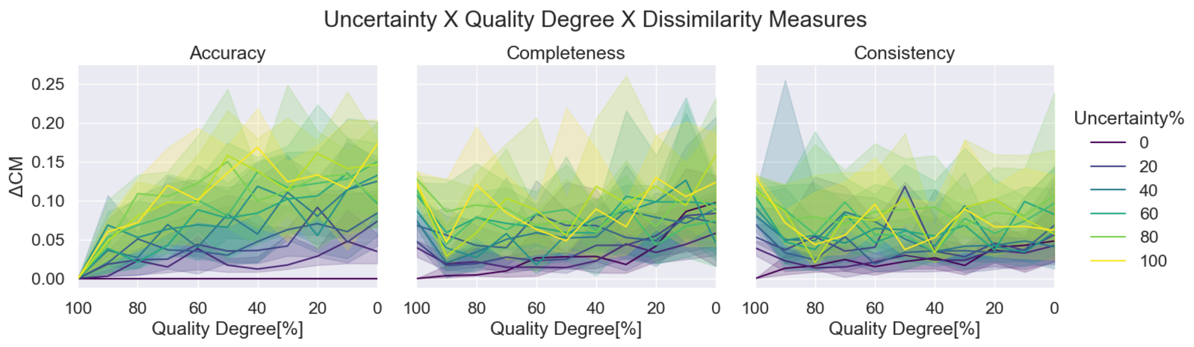


Figure 5.4: Uncertainty Assessment for Dissimilarity Measures: ΔCM

Figure 5.4 shows the Uncertainty Assessment for Dissimilarity Measures stage, with ΔCM as a measure of ambiguity of the output. The impact of uncertainty is still significant for all three dimensions of Data Quality (accuracy, completeness, and consistency) and is increasing with higher levels of uncertainty.

The impact of Data Quality is more pronounced in accuracy and completeness dimensions, where decreasing the quality results in higher slope. The impact of consistency is not as clear, with some points showing a decreasing trend in median ΔCM with decreasing quality. In the plot of the consistency in Figure 5.4 for an uncertainty of 100%, corresponding to a yellow line, the behaviour of the MAE is fluctuating especially in a range of quality degree from 80% to 30%. In this range of values, in fact, the MAE goes from 0.05 up to 0.1, then falls again around 0.05 and rises again around 0.1.

The presence of outliers, particularly in the consistency dimension, suggests that certain types of faults or anomalies have a significant effect on the dissimilarity measures. In the Figure 5.4 there is a large outlier with ΔCM equal to 0.25, for a degree of consistency of 90% and uncertainty around 50% (bluish-coloured line). This could be due to certain types of noise or outliers in the data affecting the results.

The large boundaries of different uncertainty levels and quality degrees overlapping even at low levels of uncertainty suggest that the contribution of uncertainty and Data Quality is difficult to separate and isolate. This implies that improving the Data Quality alone may not be sufficient to reduce the uncertainty in the dissimilarity measures, and more sophisticated methods or models may be needed to account for uncertainty.

5.1.3. Bootstrapping

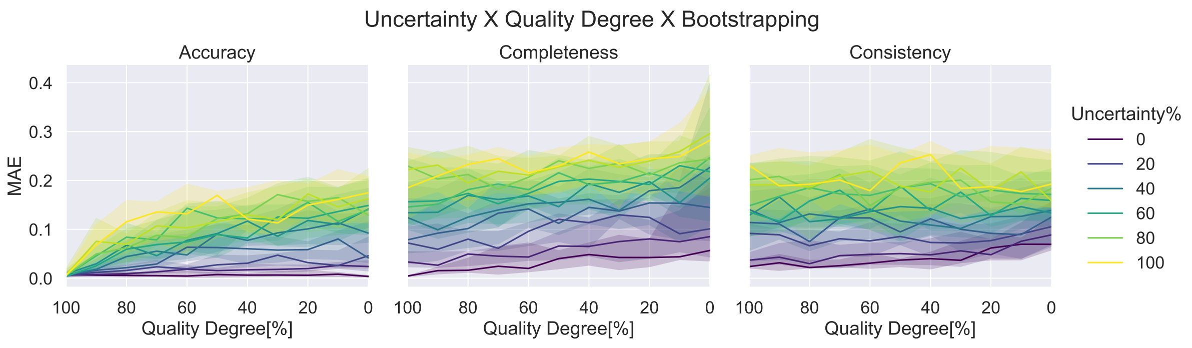


Figure 5.5: Uncertainty Assessment for Bootstrapping: MAE

Similar to the dissimilarity measures stage, the three Data Quality dimensions show increasing trends with increasing quality level and decreasing trends with increasing uncertainty level, but Figure 5.5 with a smaller value of ambiguity compared to the previous stage. The effect of uncertainty on MAE is similar to the other levels, but with a slightly smaller magnitude.

The pipeline uses bootstrapping to reduce the impact of uncertainty on the results by resampling the input curves multiple times and evaluating the fit metric on each resampled curve. This helps to reduce the impact of outliers and other sources of ambiguity on the results and provides a more robust estimate of the matching metric. Therefore, the slight decrease in magnitude observed in the uncertainty assessment for the MAE calculation at the bootstrapping stage is a desirable result, as it suggests that the bootstrapping method is effectively reducing the impact of uncertainty on the results.

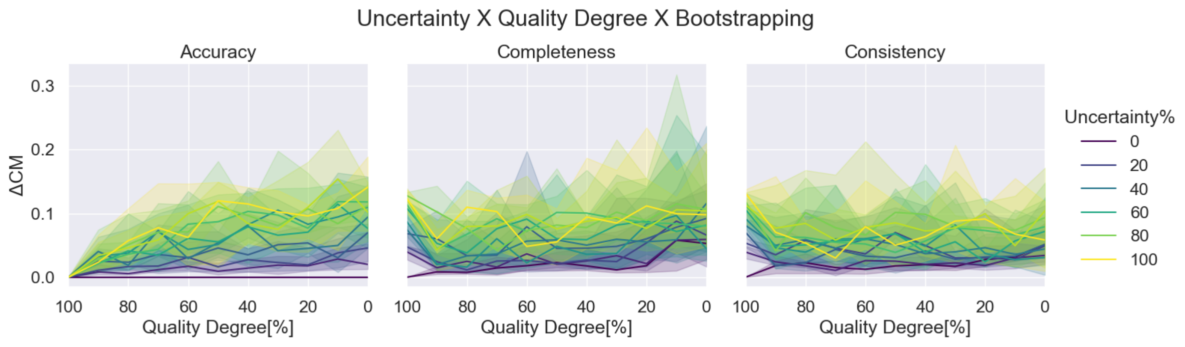


Figure 5.6: Uncertainty Assessment for Bootstrapping: ΔCM

Similar to the MAE, the ΔCM results in Figure 5.6 for the bootstrapping stage show a reduction in the magnitude of the effect compared to the dissimilarity measures stage.

There are fewer outliers in the ΔCM results, particularly in the consistency dimension, suggesting that the bootstrapping stage may be effective in reducing the impact of uncertainty on the results.

Overall, the results suggest that the bootstrapping stage can be an effective way to mitigate the effects of uncertainty and variability in the input data, leading to more consistent and reliable results in the curve matching pipeline.

5.1.4. Curve Matching Pipeline

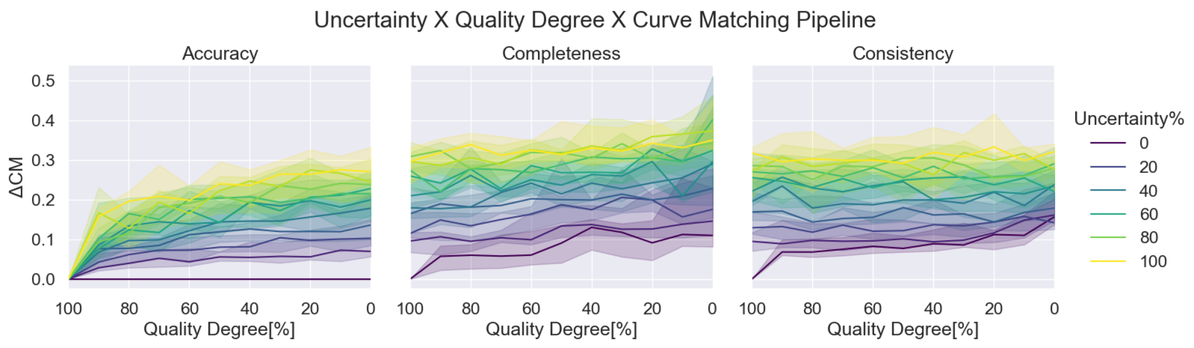


Figure 5.7: Uncertainty Assessment for Curve Matching pipeline: ΔCM

Figure 5.7 represents the results of fault injection on the Curve Matching pipeline, where ΔCM represents the ambiguity on the output of the pipeline, the Curve Matching score.

The uncertainty assessment of the ΔCM calculation on the entire curve matching pipeline showed consistent trends with the previous evaluations. The impact of uncertainty was evident in all dimensions, but the overall magnitude of the impact decreased as in the bootstrapping phase. The three dimensions showed in fact a reduced magnitude in ΔCM values along with fewer outliers and narrower bounds.

This could be related to the fact that the variations in the curve matching results were due to the injected errors rather than being influenced by the inputs or outputs of the other stages. Nevertheless, the boundaries of the uncertainty and Data Quality levels overlapped under low quality and high uncertainty conditions, indicating that the contribution of these factors was indistinguishable under these conditions. Overall, these results highlight the importance of considering uncertainty and Data Quality when evaluating curve matching performance, and the potential benefits of using the bootstrapping stage to reduce the impact of these factors on the results.

5.2. Uncertainty Propagation

The uncertainty propagation section presents a detailed analysis of the relationships between uncertainty, accuracy, completeness, and consistency in the Curve Matching pipeline. This section builds on the results of the previous section and uses MAE and ΔCM metrics to evaluate the impact of uncertainty on the quality of the results. Each plot in this section shows the uncertainty assessment results grouped for each of the injected errors, with a focus on understanding how much each error has affected the ambiguity of the results. Additionally, at the end of the section, the plots will be accompanied by a table containing the Spearman coefficient values for each of the four errors about the metrics analyzed. By examining these relationships, we gain insights into the impact of uncertainty on the quality of the results and can identify areas for improvement in the Curve Matching pipeline.

5.2.1. Functional Estimation

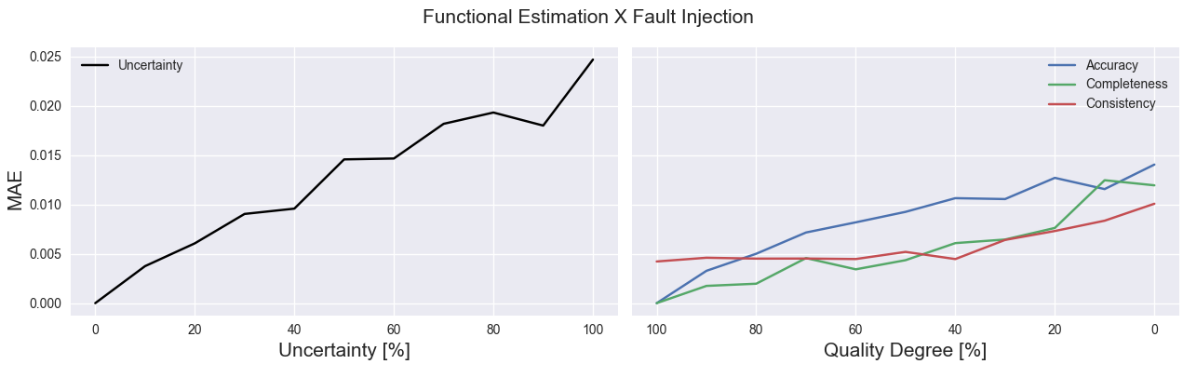


Figure 5.8: Uncertainty Propagation for Functional Estimation: MAE

The first thing that can be noticed immediately in Figure 5.8 is that in the Functional Estimation stage, the effect of the injected errors is very small in magnitude. Remembering that the plotted values are aggregated over all the experiments by the median, this means that in this case there are outliers but of a very small value compared to the others. The second possible annotation is about the uncertainty it is possible to see how this has an increasing contribution to the MAE as on the ΔCM.

It is also possible to see that the contribution of Consistency remains constant, confirming the trend of Figure 5.1, that is, where the value of MAE is more affected by uncertainty in the case of the Consistency dimension, while for the other two, it also increases as the Data Quality decreases.

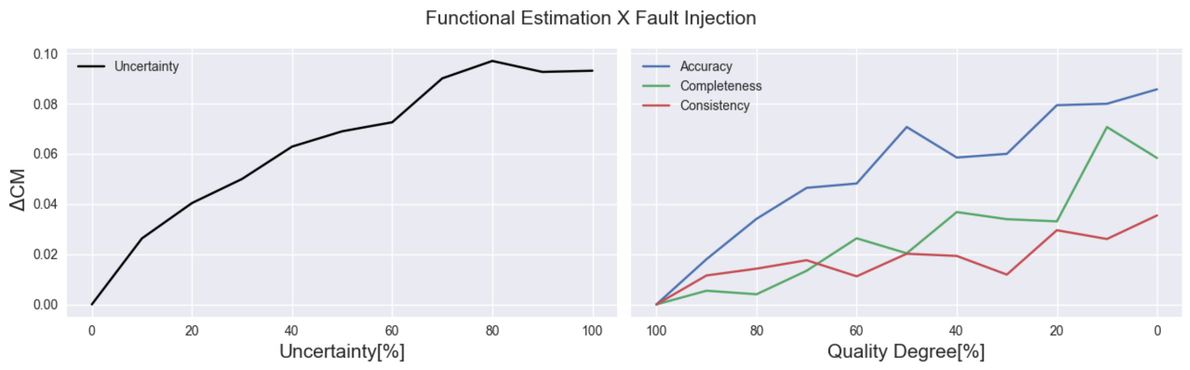


Figure 5.9: Uncertainty Propagation for Functional Estimation: ΔCM

Regarding the ΔCM in Figure 5.9, we can see that the impact of the different errors taken individually is relatively small (the contribution corresponding to the worst values is 0.1), while it is also possible to see that, compared to the MAE, in this case, the Accuracy, individually, influences more than the Completeness.

From the values of the Spearman's coefficient in Table 5.1, it is possible to see an average correlation between the analyzed variables: this means that there is a significant relationship between the analyzed metrics and the various fault injection variables, but there could be other factors or sources of variability that influence the results.

	MAE	ΔCM
Uncertainty	0.384	0.398
Accuracy	0.501	0.392
Completeness	0.62	0.479
Consistency	0.488	0.416

Table 5.1: Spearman's coefficient values for MAE and ΔCM in Functional Estimation stage.

5.2.2. Dissimilarity Measures

As for Figure 5.10, in the Dissimilarity Measures stage, fault injection has the largest impact compared to the other three stages, with a peak median of 0.175 for values corresponding to 0% completeness. It can also be seen that the contribution of Accuracy and Consistency increases steadily, while that of Completeness increases significantly as the quality levels deteriorate.

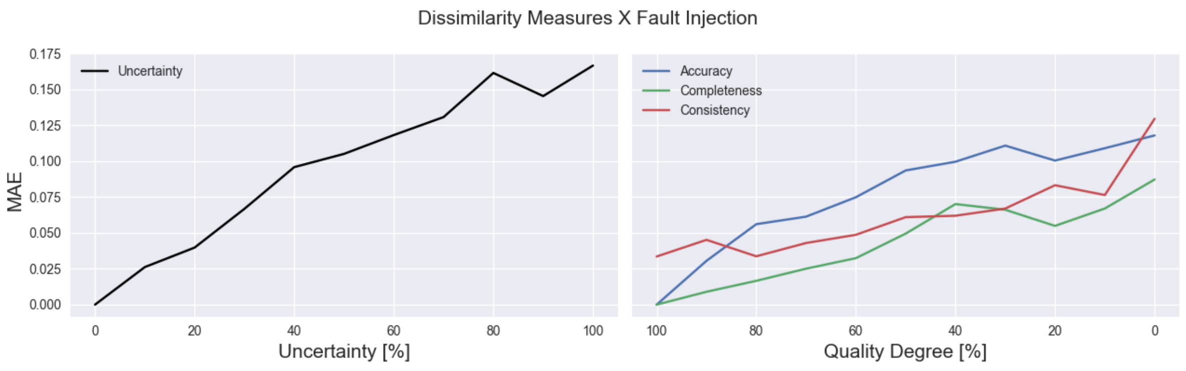


Figure 5.10: Uncertainty Propagation for Dissimilarity Measures: MAE

The situation remains the same for Figure 5.11, although the impact on ΔCM is of the same order of magnitude as for the other levels. We can also see here a uniformly increasing trend, except for Completeness, where the ΔCM value is affected quite strongly for significantly low-quality values.

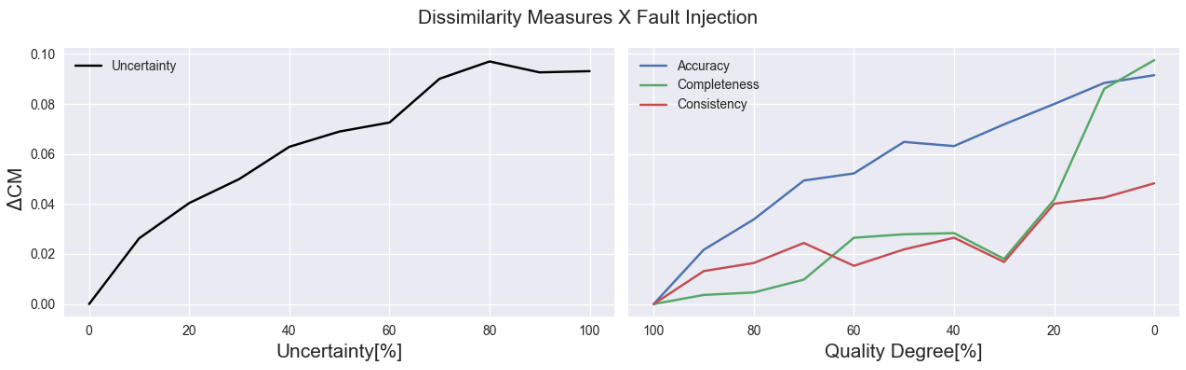


Figure 5.11: Uncertainty Propagation for Dissimilarity Measures: ΔCM

Another trend that is confirmed is that of Table 5.2, where Accuracy maintains its Spearman value around 0.35, with a higher value for ΔCM than for MAE. The values of the Data Quality dimension, on the other hand, remain moderately correlated with the two metrics, but drop in ΔCM, because, as mentioned earlier for Table 5.1, there may be other factors that affect the metrics.

5.2.3. Bootstrapping

In the bootstrapping stage, Figure 5.12 confirms the trend of reducing the impact of fault injection on the MAE stage. In particular, we can see that the impact of Uncertainty is greater than that of the Data Quality dimensions, in particular, we can see that the impact of Completeness is significantly reduced compared to the other stages.

	MAE	ΔCM
Uncertainty	0.335	0.398
Accuracy	0.448	0.392
Completeness	0.634	0.509
Consistency	0.455	0.422

Table 5.2: Spearman's coefficient values for MAE and ΔCM in Dissimilarity Measures stage.

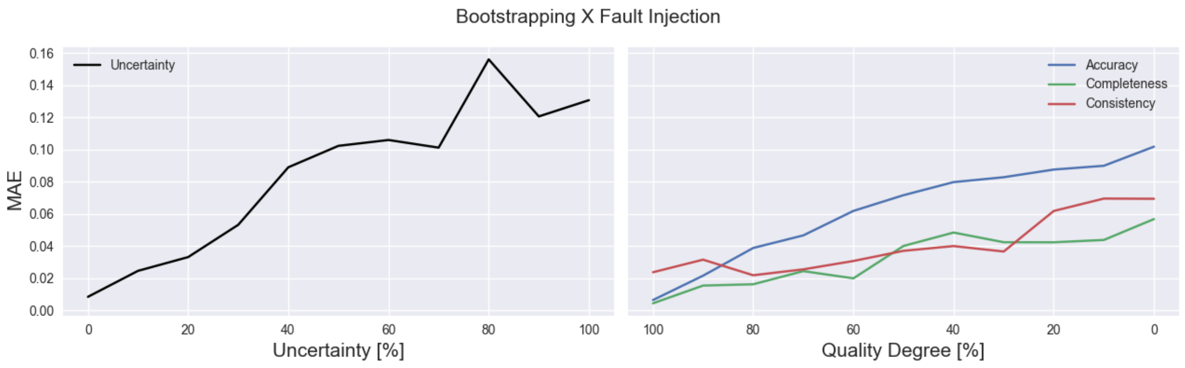


Figure 5.12: Uncertainty Propagation for Bootstrapping: MAE

On ΔCM, on the other hand, as can be seen in Figure 5.13, the stable and consistent impact with the other levels is confirmed, highlighting still the decrease in the impact of Completeness and a slight increase for that of Accuracy compared to MAE on the same level.

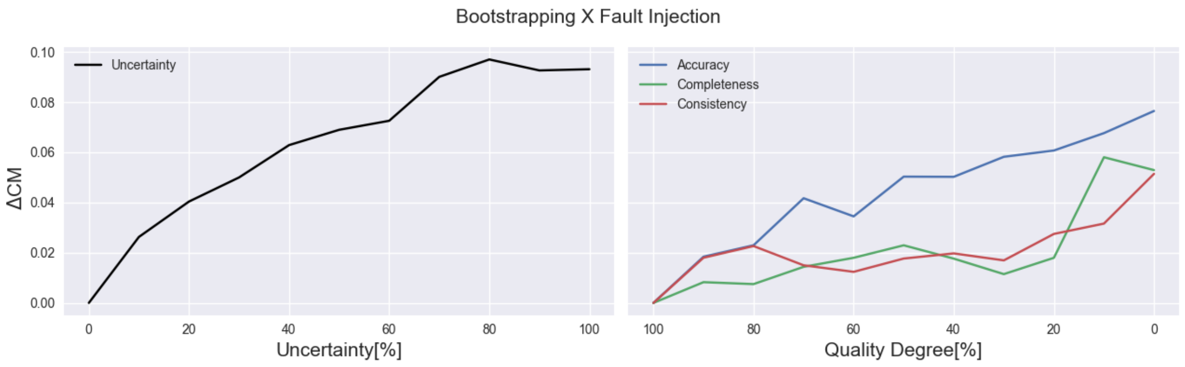


Figure 5.13: Uncertainty Propagation for Bootstrapping: ΔCM

The Spearman coefficients in Figure 5.14 confirm the trend that Uncertainty has a slightly stronger relationship with ΔCM than with MAE, while the opposite trend occurs for the other error injection variables, i.e. their impact is greater on MAE than on the final ΔCM.

	MAE	ΔCM
Uncertainty	0.367	0.398
Accuracy	0.454	0.407
Completeness	0.575	0.491
Consistency	0.493	0.478

Table 5.3: Spearman's coefficient values for MAE and ΔCM in Bootstrapping stage.

This trend holds for all stages, suggesting that the degradation at the stage level has an overall impact on ΔCM, but the uncertainty in it also depends on other factors that may affect the other stages.

5.2.4. Curve Matching Pipeline

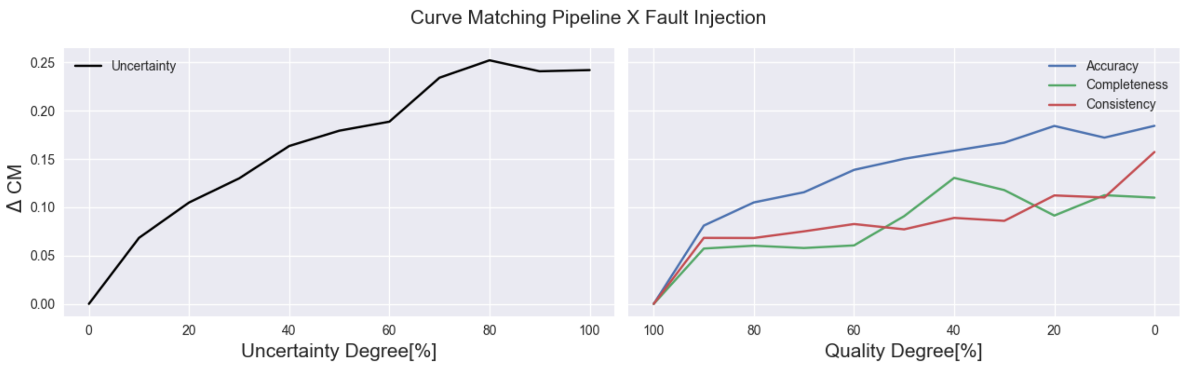


Figure 5.14: Uncertainty Propagation for Curve Matching pipeline: ΔCM

Figure 5.14 shows the impact of each individual fault injection error on the entire curve matching pipeline. First, it is possible to see the large impact on the ambiguity of the curve matching score, which is up to 0.25. This could be because the contributions of the different stages propagate through the pipeline until they result in a higher level of ambiguity. We also note that Uncertainty has the largest impact, while for the Data Quality dimensions, Accuracy has the largest impact and Completeness has the smallest impact, in contrast to the individual stages.

	ΔCM
Uncertainty	0.398
Accuracy	0.417
Completeness	0.478
Consistency	0.554

Table 5.4: Spearman's coefficient values for MAE and ΔCM in Curve Matching pipeline.

On the other hand, Table 5.4 shows the Spearman coefficients and from this, we can see that, among the individual contributions of the different errors, Uncertainty, and Accuracy have the greatest impact on ambiguity, but they are less related to it, which could depend on other factors. On the other hand, Completeness and Consistency, although they have a smaller impact at the level of magnitude, are more related to it, having a coefficient around 0.5, which shows a medium to high correlation with the final value.

5.3. Conclusions

This chapter has shown the results of the developed methodology by testing it on the Curve Matching pipeline. The methodology first involves uncertainty assessment, which is the quantification of the impact of the various faults injected through the fault injection technique into the different stages of the pipeline and into the pipeline itself. The results of fault injection allow quantification of the impact of these faults on the ambiguity of the output:

- via the MAE, which is used as a general metric for understanding ambiguity in the outputs of the different stages, and is thus invariant to the type of output returned;
- in terms of Curve Matching, by measuring the difference generated by injecting errors at both the stage and pipeline levels.

The uncertainty assessment answers the first two research questions introduced in Section 3.1. In fact, through MAE, it was possible to calculate the impact of the sources of uncertainty on the ambiguity of the outcome of each stage, regardless of the type of stage. On the other hand, by calculating the variation of the curve matching, it is possible to understand how the impact on the stage outcome is then reflected in the final pipeline outcome.

The second step is then Uncertainty Propagation, which is used to understand how much each source of uncertainty individually contributes to the ambiguities measured by the two metrics analyzed (MAE and ΔCM). Graphically, it was possible to see how uncertainty always plays an important role in all the tests performed(Figures 5.8 to 5.14), and instead how the different dimensions of Data Quality affect each analysis differently. As can be seen from the Spearman coefficient tables (Tables 5.1 to 5.4), the contribution of the error injection to the different stages measured by MAE is slightly more affected than the CM of the injected errors; this is because there are probably other sources of uncertainty within the pipeline, which, as we have seen, contains various transformations and processes that could affect the output.

Nevertheless, these analyses show an important contribution of the injected faults to the outputs of the stages and the pipeline, underlining the importance of the processes of preparation and validation of the inputs before use, but more importantly of a stage-by-stage propagation of uncertainty that is reflected in the final output of the pipeline, the Curve Matching score.

Finally, through uncertainty propagation it was possible to provide a tool to analyze the relationship between sources of uncertainty and ambiguity in the results, answering the last research question in Section 3.1.

6 | Conclusions

The methodology developed in this thesis thus provides a useful tool for understanding the propagation of uncertainty generated by various factors within a multi-stage pipeline for experimental data. Using the fault injection technique, the study identified several factors that could affect the outcome of the pipeline.

By considering the different factors involved in the fault injection, the study proposed an uncertainty assessment that aims to quantify the magnitude of ambiguity in the pipeline outputs.

Furthermore, from the results of the study, it was possible to understand how much the ambiguity on the output of the individual stage (measured by MAE) is reflected in the ambiguity of the pipeline output (measured by the ΔCM change in the final score).

Together, uncertainty assessment and uncertainty propagation made it possible to quantify the impact of sources of uncertainty for each stage, understand how this propagates on the pipeline output, and provided an analysis tool for the relationship between ambiguity and sources of uncertainty. Through this approach the research questions of Section 3.1 were answered.

One of the contributions of the study was not only the effective demonstration of the link between ambiguity and poor Data Quality but also the presence of additional unknown factors affecting the results.

For example, in Section 5.2 it is interesting how under conditions of perfect Data Quality, and in the sole presence of experimental uncertainty, the impact on ambiguity is often greatest. However, as can be seen from Spearman's ρ, this is partly due to uncertainty, otherwise, the value of the coefficient would be much closer to 1.

While the conclusions seem promising, some weaknesses should be pointed out. The fault injection technique turns out to be an efficient way to simulate errors in the experimental data handled in the pipeline, but it may not be representative of all possible sources of uncertainty in real-world scenarios. It would be beneficial to investigate other sources of uncertainty and determine how they affect the pipeline results.

In addition, the effectiveness of the methodology used is limited to the dimensions analyzed in the study. There may be other dimensions of Data Quality that generate ambiguity in the results and should be explored in future research.

Moreover, the nature of the methodology is limited by the definition of the Data Quality dimensions: an analysis performed with different definitions could lead to discrepancies.

For example, the accuracy dimension is treated as the number of perturbed points in the input experimental data set. This approach is limiting since the perturbation is always small at the scale of the data, but in real cases, it can be much more pronounced.

Finally, the conclusions presented do not guide potential solutions to the problems identified in the study. Identifying the sources of uncertainty and the ambiguity they generate is crucial, but it is equally important to explore potential solutions to these problems. Future research could focus on developing strategies to mitigate these sources of ambiguity and improve pipeline reliability.

Bibliography

[1] Barbara Pernici, Francesca Ratti, and Gabriele Scalia. About the quality of data and services in natural sciences. In *Next-Gen Digital Services. A Retrospective and Roadmap for Service Computing of the Future*, pages 236–248. Springer, 2021.

[2] M.S. Bernardi, M. Pelucchi, A. Stagni, L.M. Sangalli, A. Cuoci, A. Frassoldati, P. Secchi, and T. Faravelli. Curve matching, a generalized framework for models/experiments comparison: An application to n-heptane combustion kinetic mechanisms. *Combustion and Flame*, 168:186–203, 2016.

[3] Camilla Sancricca. Context aware data preparation. Master's thesis, Politecnico di Milano, 2021.

[4] Leonardo Febbo. An adaptive approach for supporting data preparation phase. Master's thesis, Politecnico di Milano, 2021.

[5] Edoardo Ramalli, Timoteo Dinelli, Andrea Nobili, Alessandro Stagni, Barbara Pernici, and Tiziano Faravelli. Automatic validation and analysis of predictive models by means of big data and data science. *Chemical Engineering Journal*, 454:140149, 2023.

[6] Edoardo Ramalli, Barbara Pernici, et al. Know your experiments: interpreting categories of experimental data and their coverage. In *CEUR WORKSHOP PROCEEDINGS*, volume 2929, pages 27–33. CEUR Workshop Proceedings, 2021.

[7] Thomas Lee Paez. Introduction to model validation. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2008.

[8] Leonard E Schwer. Guide for verification and validation in computational solid mechanics. 2009.

[9] Carsten Olm, István Gy Zsély, Róbert Pálvölgyi, Tamás Varga, Tibor Nagy, Henry J Curran, and Tamás Turányi. Comparison of the performance of several recent hydrogen combustion mechanisms. *Combustion and Flame*, 161(9):2219–2234, 2014.

[10] Carsten Olm, István Gy Zsély, Tamás Varga, Henry J Curran, and Tamás Turányi.

Comparison of the performance of several recent syngas combustion mechanisms. *Combustion and Flame*, 162(5):1793–1812, 2015.

[11] Edoardo Ramalli and Barbara Pernici. Knowledge graph embedding for experimental uncertainty estimation. *Information Discovery and Delivery*, 2023.

[12] Edoardo Ramalli, Gabriele Scalia, Barbara Pernici, Alessandro Stagni, Alberto Cuoci, and Tiziano Faravelli. Data ecosystems for scientific experiments: managing combustion experiments and simulation analyses in chemical engineering. *Frontiers in big Data*, page 67, 2021.

[13] J. O. Ramsay and B. W. Silverman. *Smoothing functional data with a roughness penalty*, pages 81–109. Springer New York, New York, NY, 2005.

[14] J. O. Ramsay. *Functional Data Analysis.* John Wiley & Sons, Ltd, 2004.

[15] Tino Kluge. Cubic spline interpolation in c++, 2021.

[16] Wikipedia. Spline (mathematics), 2022.

[17] Carl de Boor. *Spline Basics*, pages 141–163. 01 2002.

[18] John Harrison. Without loss of generality. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *Theorem Proving in Higher Order Logics*, pages 43–59, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[19] Christopher Z Mooney, Christopher F Mooney, Christopher L Mooney, Robert D Duval, and Robert Duvall. *Bootstrapping: A nonparametric approach to statistical inference.* Number 95. sage, 1993.

[20] Antonio Cuevas, Manuel Febrero, and Ricardo Fraiman. On the use of the bootstrap for estimating functions with functional data. *Computational Statistics & Data Analysis*, 51(2):1063–1074, 2006.

[21] François M Hemez and Scott W Doebling. Model validation and uncertainty quantification. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2000.

[22] Ramesh Rebba, Sankaran Mahadevan, and Shuping Huang. Validation and error estimation of computational models. *Reliability Engineering & System Safety*, 91(10-11):1390–1397, 2006.

[23] Joshua Mullins, You Ling, Sankaran Mahadevan, Lin Sun, and Alejandro Strachan. Separation of aleatory and epistemic uncertainty in probabilistic model validation. *Reliability Engineering & System Safety*, 147:49–59, 2016.

[24] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[25] Michael Frenklach, Andrew Packard, Pete Seiler, and Ryan Feeley. Collaborative data processing in developing predictive models of complex reaction systems. *International journal of chemical kinetics*, 36(1):57–66, 2004.

[26] Michael Frenklach, Andrew Packard, and Pete Seiler. Prediction uncertainty from models and data. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 5, pages 4135–4140. IEEE, 2002.

[27] Pete Seiler, Michael Frenklach, Andrew Packard, and Ryan Feeley. Numerical approaches for collaborative data processing. *Optimization and Engineering*, 7(4):459–478, 2006.

[28] Ryan Feeley, Michael Frenklach, Matt Onsum, Trent Russi, Adam Arkin, and Andrew Packard. Model discrimination using data collaboration. *The Journal of Physical Chemistry A*, 110(21):6803–6813, 2006.

[29] Michael Frenklach, Andrew Packard, and Ryan Feeley. Optimization of reaction models with solution mapping. *Comprehensive Chemical Kinetics*, 42:243–291, 2007.

[30] Michael Frenklach, Andrew Packard, Gonzalo Garcia-Donato, Rui Paulo, and Jerome Sacks. Comparison of statistical and deterministic frameworks of uncertainty quantification. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):875–901, 2016.

[31] Joshua Mullins, You Ling, Sankaran Mahadevan, Lin Sun, and Alejandro Strachan. Separation of aleatory and epistemic uncertainty in probabilistic model validation. *Reliability Engineering & System Safety*, 147:49–59, 2016.

[32] Ramesh Rebba and Sankaran Mahadevan. Computational methods for model reliability assessment. *Reliability Engineering & System Safety*, 93(8):1197–1207, 2008.

[33] Unmeel Mehta, Dean Eklund, Vicente Romero, Jeffrey Pearce, and Nicholas Keim. Simulation credibility. subtitle: Advances in verification, validation, and uncertainty quantification. 11 2016.

[34] Haldun Akoglu. User's guide to correlation coefficients. *Turkish journal of emergency medicine*, 18(3):91–93, 2018.

[35] Carlo Batini, Daniele Barone, Michele Mastrella, Andrea Maurino, and Claudio Ruffini. A framework and a methodology for data quality assessment and monitoring. pages 333–346, 01 2007.

[36] Barbara Pernici, Francesca Ratti, and Gabriele Scalia. *About the Quality of Data and Services in Natural Sciences*, pages 236–248. Springer International Publishing, Cham, 2021.

[37] P. Mathias, A. Soto, Ljudmila Fele Žilnik, Jean-Charles Hemptinne, Ala Bazyleva, and J. Abildskov. Data quality and assessment, validation methods and error propagation through the simulation software: Report from the round-table discussion at the 10th world congress of chemical engineering in barcelona (october 1-5, 2017). *Chemical engineering research & design : transactions of the Institution of Chemical Engineers*, 137, 09 2018.

[38] Carlo Batini and Monica Scannapieco. *Data Quality: Concepts, Methodologies and Techniques.* 01 2006.

[39] Richard Y. Wang and Diane M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, 12(4):5–33, 1996.

[40] Divyakant Agrawal, Philip Bernstein, Elisa Bertino, Susan Davidson, Umeshwas Dayal, Michael Franklin, Johannes Gehrke, Laura Haas, Alon Halevy, Jiawei Han, et al. Challenges and opportunities with big data 2011-1. 2011.

[41] Carlo Bono, Mehmet Oğuz Mülâyim, Cinzia Cappiello, Mark James Carman, Jesus Cerquides, Jose Luis Fernandez-Marquez, Maria Rosa Mondardini, Edoardo Ramalli, and Barbara Pernici. A citizen science approach for analyzing social media with crowdsourcing. *IEEE Access*, 11:15329–15347, 2023.

[42] Jason Osborne. Notes on the use of data transformations. *Practical assessment, research, and evaluation*, 8(1):6, 2002.

[43] Maria Grazia Fugini, Barbara Pernici, and Filippo Ramoni. Quality analysis of composed services through fault injection. *Information Systems Frontiers*, 11:227–239, 2009.

[44] Jukka K Nurminen, Tuomas Halvari, Juha Harviainen, Juha Mylläri, Antti Röyskö, Juuso Silvennoinen, and Tommi Mikkonen. Software framework for data fault injection to test machine learning systems. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 294–299. IEEE, 2019.

[45] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[46] Wes McKinney et al. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011.

[47] Sandro Tosi. *Matplotlib for Python developers.* Packt Publishing Ltd, 2009.

[48] Michael L Waskom. Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.

[49] Igor Stančin and Alan Jović. An overview and comparison of free python libraries for data mining and big data analysis. In *2019 42nd International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 977–982. IEEE, 2019.

List of Figures

List of Tables

Acronyms

SSE Sum Squared Error. 14, 19

UQ Uncertainty Quantification. 7, 26–28

V&V Verification & Validation. 7

WLOG Without Loss Of Generality. 21