Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria

Master of Science in Automation and Control Engineering

Projection- vs. Selection-based Model Reduction For Emulation Modeling in Water Resources Planning and Management Problems

Supervisor:
Prof. Andrea Castelletti
SUTD Co-Supervisor:
Prof. Stefano Galelli
Co-supervisor:
Matteo Giuliani, PhD

Master Thesis by:
Ahmad Alsahaf
Matr. 786025

Academic Year 2014-2015

Acknowledgments

This thesis was completed under the supervision of Andrea Castelletti and Matteo Giuliani of Politecnico di Milano, and Stefano Galelli of The Singapore University of Technology and Design. I would like to thank them all for the opportunity they gave me to work on such an exciting project, and for their continuous support and guidance.

Special thanks goes to Stefano Galelli for giving me the opportunity to work on this project at the Singapore University of Technology and Design, and for the staff of SUTD for being such great hosts.

I would like to thank my friends and colleagues in Milan for all the good times we had, and for making life away from home easier and more enjoyable.

My love and gratitude goes to my parents for the sacrifices they made on my behalf. Their strength and patience continue to inspire me. I would also like to thank my sister and her lovely family for being in my life.

Contents

List of Figures

List of Tables

Abstract

Projection-based model reduction is one of the most popular approaches used for the identification of reduced-order models (emulators). It is based on the idea of using data samples, or snapshots from the original model to project it onto a lower dimensional subspace that captures the majority of the variation of the original model. Yet, this approach may unnecessarily increase the complexity of the emulator, especially when only a few state variables of the original model are relevant with respect to the output of interest. This is the case of complex hydro-ecological models, which typically account for a variety of water quality processes. On the other hand, selection-based model reduction uses the information contained in the snapshots to select the state variables of the original model that are relevant with respect to the emulator's output, thus allowing for model reduction. This provides a better trade-off between fidelity and model complexity, since the irrelevant and redundant state variables are excluded from the model reduction process. In this thesis, these issues are addresses by performing an experimental comparison between a selection-based method, i.e. Recursive Variable Selection (RVS), and a projection-based method, i.e. Principal Component Analysis (PCA), and two variants of the latter, namely, Sparse PCA and Weighted PCA. The comparison is performed on the reduction of DYRESM-CAEDYM, a 1D hydro-ecological model used to describe the in-reservoir water quality conditions of Tono Dam, an artificial reservoir located in western Japan. Experiments on three different output variables (i.e. released water temperature, released sediments, and in-reservoir chlorophyll-a concentration) show that RVS achieves the same fidelity as PCA, while reducing the number of state variables in the emulators. Moreover, Sparse PCA and Weighted PCA were found to mitigate some of the disadvantages of ordinary PCA, thus increasing the accuracy and improving physical intepretability.

Riassunto

Riduzione modello basato proiezione è uno degli approcci più popolari utilizzati per l'identificazione dei modelli in ordine (emulatori). Si basa sull'idea di utilizzando campioni di dati, o snapshots dal modello originale per proiettarla su un minore sottospazio tridimensionale che cattura la maggior parte della variazione dell'originale modello. Tuttavia, questo approccio può aumentare inutilmente la complessitÃ dell'emulatore, soprattutto quando solo pochi variabili di stato del modello originale sono rilevanti rispetto all'uscita di interesse. Questo è il caso del complesso idro-ecologica modelli, che in genere rappresentano una varietà di processi di qualità delle acque. D'altronde, la riduzione modello di selezione basata utilizza le informazioni contenute nel snapshots per selezionare le variabili di stato del modello originale che sono pertinenti con rispetto all'uscita dell'emulatore, consentendo così la riduzione del modello. Ciò fornisce una migliore trade-off tra infedeltà e complessità del modello, dal momento che l'irrilevante e variabili di stato ridondanti sono esclusi dal processo di riduzione del modello. In questo tesi, questi problemi sono indirizzi eseguendo un confronto sperimentale tra un metodo di selezione basata, cioè Recursive Variable Selection (RVS), e una proiezione basata metodo, cioè Principal Component Analysis (PCA), e due varianti di quest'ultimo, vale a dire, Sparse PCA e Weighted PCA. Il confronto viene effettuato sulla riduzione di DYRESM-CAEDYM, un modello idro-ecologico 1D utilizzato per descrivere la in-serbatoio condizioni di qualità dell'acqua di Tono Dam, un serbatoio ciale arti situato in Giappone occidentale. Esperimenti su tre variabili di uscita erente (acqua cioè rilasciata temperatura, rilasciato sedimenti, e in-serbatoio concentrazione di clorofilla a) spettacolo RVS che raggiunge lo stesso infedeltà come PCA, riducendo il numero di stato variabili nei emulatori. Inoltre, Sparse PCA e Weighted PCA sono stati trovati per mitigare alcuni degli svantaggi di ordinaria PCA, aumentando così la precisione e migliorando intepretability fisica.

Introduction

Computer simulation of process-based models has been used extensively and with success to describe complex physical systems and phenomena. These models have been used in different scientific and engineering disciplines, ranging from atmospheric science to water resource engineering. However, as the complexity and dimensionality of these models increase, the computational and storage requirements also increase, oftentimes to unfeasible levels (Washington et al., 2009). Model-order reduction (or emulation modeling) is often used to mitigate this problem. The goal of model-order reduction is to create a lower-order model, also known as a *surrogate model* or *emulator*, which mimics the behavior of the full-sized model as closely as possible while alleviating significantly the computational and storage burden.

In addition to the underlying complexity of certain mathematical models, an even greater demand of computational and storage resources is often caused when dealing with problems that require a large number of model evaluations such as: optimal planning, optimal control or management, model structure identification, data assimilation and forecasting, and sensitivity analysis. Problems of this kind can benefit largely from model-reduction.

A reduced model can be derived by simplifying the structure of the process-based models, or it can be identified on the basis of the data obtained via simulation. This study is concerned with the latter type. More specifically, it deals with Dynamic Emulation Modeling (DEMo), a model-reduction approach that preserves the dynamic nature of the original process-based model (Castelletti et al., 2012a).

For the purpose of this study, model-reduction is classified into two broad categories: Projection-based, and Selection-based model reduction techniques. Projection-based approaches project a high-dimensional space vector of the process-based model onto a lower dimensional space. On the contrary, in the selection-based approaches, a subset of the states from the process-based model is selected based on its relevance to the output of the emulator, resulting in a model of lower dimensionality.

The aim of this study is to assess the effectiveness of the two approaches, and to compare them in terms of the fidelity, complexity, and physical interpretability of the resulting reduced models. The comparison is performed on the reduction of DYRESM-CAEDYM, a 1D hydro-ecological model used to describe the in-reservoir water quality conditions of Tono Dam, an artificial reservoir located in western Japan. This model is a suitable candidate for model-reduction, as it has been employed in the design of the multi-purpose operations of the dam (Castelletti et al., 2014), which is a computationally demanding process.

The widely used method of *Principal Component Analysis* (or PCA) is chosen to represent projection-based approaches in this thesis. PCA has has been widely used in the environmental modeling literature. Examples include ground water modeling (e.g. Winton et al., 2011), oceanography (e.g. van der Merwe et al., 2007), and water quality control (e.g. Xu et al., 2013). For selection-based approaches, the method known as *Recursive Variable Selection* (RVS) is selected. RVS was introduced by Castelletti et al. (2012b) and used for the emulation modeling of Tono Dam.

Principal Component Analysis works by projecting all the original variables into a new set of variables (principal components). This projection is defined in such a way that the first principal component explains the largest amount of variance from the original variables, and each succeeding component in turn explains the highest variance possible under the constraint that it is orthogonal to the preceding components. On the other hand, Recursive Variable Selection selects, through a correlation analysis, the most suitable variables from the original variables to explain the output of interest, while discarding the rest of the variables.

This discrepancy between the two approaches may give PCA an advantage over RVS in terms of overall reduction of the dimensionality. However, this comes at the price of increased ambiguity of the reduced model, because all original variables are still present. Therefore, RVS becomes favorable in applications where physical interpretability must be preserved.

In order to achieve a high reduction of dimensionality without losing physical interpretability, two additional model-reduction techniques are explored, namely, Sparse Principal Component Analysis (SPCA), and Weighted Principal Component analysis (WPCA).

Overview

- **Chapter 1:** The first chapter gives a general overview of model-order reduction, and introduces Dynamic Emulation Modeling, or the DEMo procedure (Castelletti et al., 2012b), which is a unified, six-step procedure upon which the model-reduction exercises in this study will be based.

- **Chapter 2:** The second chapter consists of the theoretical backgrounds of the methodologies used in this thesis for model-reduction, beginning with selection-based approaches and ending with projection-based ones.

- **Chapter 3:** This chapter describes the case study; the Tono dam system, including: (i) a description of DYRESM-CAEDYM, the 1D spatially-distributed model used to describe the in-reservoir hydrological and ecological processes, (ii) a description of the input/output variables that are used in the emulation exercise, and (iii) the formulation of the optimal control problem of the water system.

- **Chapter 4:** The fourth and final chapter presents the results of implementing the DEMo procedure on the Tono Dam system with the competing approaches, and concludes with a comparison between the obtained emulators in terms of accuracy, complexity, and physical interpretability.

Chapter 1

Overview of Model-order Reduction

The goal of this chapter is to give a general overview of model-order reduction or emulation modeling, and to introduce a general framework upon which the competing methodologies will be based.

It is worth noting that techniques compared in this thesis are *data-driven* model-reduction techniques, i.e., they are identified on a data-set of inputs, state-variables, and outputs generated by simulations of the process-based, full-sized model. Therefore, they do not rely on the original structure of the model, nor do they attempt to manipulate it. Furthermore, these techniques fall under Dynamic Emulation Modeling techniques (DEMo), i.e., they aim at preserving the dynamic behavior of the original process-based model. The framework used in this study is the one described by Castelletti et al. (2012a), which provides a unified, six-step procedure that can be followed for both structure-based or data-based reduction techniques.

1.1 Introduction

An emulation model, or emulator, is a low-order approximation of the physically-based model that can be substituted for it in order to solve a high resource-demanding problems. Such a model can be derived by simplifying the physically-based model structure, or it can be identified on the basis of the response data produced by simulating this large model with carefully selected input perturbations. Dynamic Emulation Modeling (DEMo) is a special type of model complexity reduction, in which the dynamic nature of the original physically-based model is preserved, with consequent advantages in a wide range of problems, such as optimal control.

There is a large number of applications that benefit form emulation modeling including: management of environmental resources, fluid dynamics, image processing, and biological systems. As the number keeps increasing, it becomes useful to describe of a unified design framework for the different strategies of complexity reduction and emulation. The next section describes the steps of such procedure. These steps can be used to formulate any emulation problem. In particular, it is used in this study for the Tono Dam system water system.

1.2 Framing The Problem

1.2.1 The System $\mathcal{E}$

Let's consider a large environmental system $\mathcal{E}$, whose state $\mathcal{X}(t, s)$ varies in a time-space domain $\mathcal{T} \times \mathcal{S}$. The system is affected by a time-varying, often distributed in space, exogenous driver $\mathcal{W}(t, s)$.

The output $\mathcal{Y}(t)$ is generally, but not necessarily, lumped and is constituted by the variables that are relevant to the analyst: it usually comprises few variables but it can sometimes be distributed in space and coincide with the whole state.

Engineering applications are often related to the problem of controlling or managing the dynamics of $\mathcal{X}(t, s)$ and $\mathcal{Y}(t)$ through a sequence of decisions, periodically repeated over the whole system's life. In this case, a control vector $\mathbf{u}_t$ is applied to $\mathcal{E}$ [1] at discrete time instants, according to a decision time-step. The system $\mathcal{E}$ can also be affected by a vector $\mathbf{v}$ of planning decisions that are normally not changed over the whole life of the system.

1.2.2 The Model $\mathcal{M}$

The scientific approach to environmental systems modeling normally exploits physical knowledge about the dynamic behavior of the system $\mathcal{E}$ to build more or less sophisticated process-based models that reproduce the perceived reality as well as possible. These models can be separated into two, broad families: physically-based and conceptual models (Wheater and Beven, 1993).

[1] It is assumed that system $\mathcal{E}$ is controllable. Operationally, the controllability of $\mathcal{E}$ must be verified before the emulation modeling exercise.

Physically-based models.

The system $\mathcal{E}$ is described by a large, generally nonlinear, dynamic model, normally defined in $\mathcal{T} \times \mathcal{S}$ by a set of partial differential equations (PDE). These equations describe the evolution of the system state $\mathcal{X}(t, s)$ and output $\mathcal{Y}(t)$ in response to external forcing $\mathcal{W}(t, s)$ (either deterministic or stochastic) and control $\mathbf{u}_t$.

Conceptual Models.

1. **Continuous-time.** Although a PDE model could be used, the system $\mathcal{E}$ is normally described by a continuous-time, non-linear model, formulated as a system of ordinary differential equations, based on a conceptualization and simplification of the physical laws describing the system dynamics.

   $$\dot{\mathbf{X}}(t) = \mathbf{F}\left(t, \mathbf{X}(t), \mathbf{W}(t), \mathbf{u}(t), \mathbf{v}|\boldsymbol{\Theta}\right) \tag{1.1a}$$

   $$\mathbf{Y}(t) = \mathbf{H}\left(t, \mathbf{X}(t), \mathbf{W}(t), \mathbf{u}(t), \mathbf{v}|\boldsymbol{\Theta}\right) \tag{1.1b}$$

   where the information content of $\mathcal{X}(t, s)$ and of $\mathcal{W}(t, s)$ is lumped into the vectors $\mathbf{X}(t)$ and $\mathbf{W}(t)$, and $\mathcal{Y}(t) = \mathbf{Y}(t)$ while $\mathbf{F}(\cdot)$ is a generally non-linear, time-variant, vector function that models the dynamics of $\mathbf{X}(t)$; $\mathbf{H}(\cdot)$ is a generally non-linear, possibly time-variant, output transformation function, $\mathbf{v}$ are the planning decisions; and $\boldsymbol{\Theta}$ is the vector of the model parameters.

2. **Discrete-time.** The system $\mathcal{E}$ is described by a discrete-time, non-linear model, formulated as a system of finite-difference equations:

   $$\dot{\mathbf{X}}_{t+1} = \mathbf{F}_t\left(\mathbf{X}_t, \mathbf{W}_t, \mathbf{u}_t, \mathbf{v}|\boldsymbol{\Theta}\right) \tag{1.2a}$$

   $$\mathbf{Y}_t = \mathbf{H}_t\left(\mathbf{X}_t, \mathbf{W}_t, \mathbf{u}_t, \mathbf{v}|\boldsymbol{\Theta}\right) \tag{1.2b}$$

   where the information content of $\mathcal{X}(t, s)$, $\mathcal{W}(t, s)$, and $\mathcal{Y}(t)$ is now sampled, typically at a uniform sampling interval Δt, and transformed into the sampled data vectors $\mathbf{X}_t$, $\mathbf{W}_t$, and $\mathbf{W}_t$.

   The spatial aspects are normally defined by the state and exogenous driver vectors $\mathbf{X}_t$ and $\mathbf{W}_t$, which are defined at different spatial locations. In the presence of $\mathbf{u}_t$, the sampling time step is generally assumed equal to the decision time step, otherwise, only the former exists and is related to the frequency of observations available or, when this is not limiting, based on the problem at hand.

The function $\mathbf{F}_t(\cdot)$ is a generally non-linear, time-variant, vector function that models the dynamics of $\mathbf{X}_t$; $\mathbf{H}_t(\cdot)$ is a generally non-linear, possibly time-variant, output transformation function, and $\boldsymbol{\Theta}$ is a vector of the model parameters.

When a physically-based (or a conceptual continuous-time) model is adopted, an explicit scheme is commonly used for its numerical solution. In practice, this requires the discretization of the time-space domain $\mathcal{T} \times \mathcal{S}$ (or simply the time domain $\mathcal{T}$) with an appropriate grid. In this way, the original continuous-time model is, *de facto*, transformed into a discrete-time model of the form (1.2a). When the original model is physically-based, all the variables, apart from $\mathbf{u}_t$ and $\mathbf{v}$, which are not spatially distributed, have a dimensionality equal to their original dimensionality times the cardinality of the space discretization grid. When the original model is conceptual, the dimensionality of all the variables is unchanged.

In conclusion, whatever the process-based model adopted, a distinctive feature of the model $\mathcal{M}$ is the large dimensionality of the state, exogenous driver, and parameter vectors which, on one hand, is required for a detailed description of the processes in $\mathcal{E}$ but, on the other hand, makes it computationally too intensive for those problems that require hundreds or thousands of model runs.

1.2.3 The Problem $\mathcal{P}$

Assume that we have a model $\mathcal{M}$ together with a certain defined problem $\mathcal{P}$. For this model, according to its complexity, a full and proper statistical estimation or 'calibration' of its parameters may not be feasible, so that this has been performed as well as possible. Depending on $\mathcal{P}$, our interest might be either in the trajectory of $\mathbf{Y}_t$, or in a functional $J(\cdot)$ of this trajectory. Problems $\mathcal{P}$ are generally known and classified in the following categories:

- ***Optimal management and planning*** In optimal management (or optimal control) problems, the purpose is to design the feedback control policy $\mathcal{P}$ that maximizes the function $J(\cdot)$. Instead, in optimal planning, the vector $\mathbf{v}$ that maximizes $J(\cdot)$ has to be determined. Depending on the dimensionality of $\mathbf{v}$, the size of the associated feasibility domain, and the complexity of the functional and constraint shape, the algorithms available to solve optimal planning problems (basically, simulation-based optimization algorithms) are hardly usable with large process-based models. The topic has been widely explored in the environmental modeling literature; recent examples include air quality planning, water quality planning, water distribution networks, water supply system, etc.

- ***Model Diagnostics*** The selection and use of diagnostic measures are important elements in the modeling exercise, both within the model building itself (i.e., as a fundamental preliminary step prior to the practical application of the model) and in analyzing the model-based results used to solve a problem $\mathcal{P}$. In the first case, diagnostic tools are used to test or validate hypotheses and parametrizations against available observations; or with respect to some desirable or plausible behavior of model outputs of interest. In the second case, diagnostic tools can be used to assess the robustness of results (e.g. in control or planning problems) and make them more transparent to users, stakeholders and policy-makers. Diagnostic problems arising when evaluating the model $\mathcal{M}$ are summarized below.

  - *Model structure identification.* The large physically-based model structure is usually specified by the modeler's choice of a specific model form and order that best represent the system under analysis. After the model structure is defined, however, the model should undergo a thorough identification, estimation (calibration) and validation analysis, before using it for practical applications. The relation between data and parameters $\boldsymbol{\Theta}$ must be considered: an increase in model complexity is indeed reflected on an increase in the number of parameter $\boldsymbol{\Theta}$ to be defined and calibrated. This can easily lead to over-parametrization and non-uniqueness (i.e., the presence of multiple models or parameter sets that have equally acceptable fits to observational data). To avoid this problem, statistical techniques can be used to assess the discrepancy between the data information content and the number of parameters to be calibrated.
  - *Sensitivity analysis.* Uncertainty analysis aims at quantifying the uncertainty associated with the model output or a functional $J(\cdot)$ thereof,

given some ‘prior’ uncertainty, usually based on expert judgment, or after parameter estimation (calibration) has been completed. Uncertainty quantification should always be accompanied by a sensitivity analysis (Saltelli et al., 2000). Performing an uncertainty and sensitivity analysis involves the use of Monte Carlo sampling and performing a large number of model evaluations by varying model parameters $\mathbf{\Theta}$. In the presence of large, complex models, this is simply not affordable and the use of emulators often represents the only possible solution to this kind of problem.

  - *Data assimilation.* IF some or all of the outputs $\mathbf{Y}_t$ are being monitored on a regular basis, it is often possible to combine these measurements with the model $\mathbf{X}_t$ predictions to produce real-time estimates and forecasts of the state variables.

- ***Simulation*** The model $\mathcal{M}$ is the tool for analyzing the behavior of the system $\mathcal{E}$ under different trajectories of the exogenous driver $\mathbf{W}_t$, the control variable $\mathbf{u}_t$ and alternatives of $\mathbf{u}^p$. Simulation analysis, often referred to as scenario analysis, what-if analysis or policy[2] simulation, can be seen as an elementary and necessary step in almost all the above mentioned categories.

Real-world studies and applications often deal with more complicated problems that can be seen as a combination of the above mentioned problems. In all these cases, the solution of any problem $\mathcal{P}$ is practically unfeasible due to the large computational requests. As the core of the difficulty stands in the dimensionality of model $\mathcal{M}$, the natural solution is to identify a reduced model that accurately emulates the output $\mathbf{Y}_t$, or the functional $J(\cdot)$, of model $\mathcal{M}$, but with a dimensionality such that problem $\mathcal{P}$ can be solved. The reduced model is called an emulation model and it substitutes model $\mathcal{M}$ in problem $\mathcal{P}$: this replacement is possible because some processes described by the process-based model are more significant than others with respect to $\mathbf{Y}_t$ or $J(\cdot)$.

[2] A periodic sequence of control laws, which, given the current state $\mathbf{X}_t$ of the system $\mathcal{E}$ at each time instant t, suggests the optimal control to be adopted.

1.3 Complexity Reduction

As mentioned in the previous section, the emulator m, once identified, can be used in place of $\mathcal{M}$ in solving the problem $\mathcal{P}$. Depending on whether the purpose of the emulation modeling is to reproduce $\mathbf{Y}_t$ or $J(t)$, the techniques available in the literature can be re-framed into two methodological approaches: Dynamic Emulation modeling (DEMo) and non-dynamic emulation modeling. The The emulator neither modifies nor improves the conceptual features of the model $\mathcal{M}$; it simply makes it computationally more efficient in solving the problem $\mathcal{P}$. Hence, the consistency of an emulator is simply inherited from $\mathcal{M}$, which has to provide a meaningful and reliable representation of the system $\mathcal{E}$ for the range of inputs (exogenous drivers, control and planning variables) and parameters specified by the user.

1.3.1 Dynamic Emulation Modeling (DEMo)

Dynamical Emulation modeling provides a simplified description of the model $\mathcal{M}$ that preserves its dynamical nature. For this reason, the target of DEMo is to construct an approximation $\mathbf{y}_t$ of the model $\mathcal{M}$ output $\mathbf{Y}_t$ (such that $\mathbf{y}_t \sim \mathbf{Y}_t$) by adopting a considerably smaller number of variables (states $\mathbf{x}_t$ and/or exogenous drivers $\mathbf{w}_t$) and, possibly, parameters $\mathbf{\Theta}$. The rationale behind this dimensionality reduction is that some of the processes described by the model $\mathcal{M}$ are more significant than others in affecting $\mathbf{Y}_t$, so that any simpler model that describes, as well as possible, only these processes and ignores the others can be considered as operationally equivalent to the model $\mathcal{M}$ with respect to problem $\mathcal{P}$. The identified dynamic emulator m is such that it is less computationally intensive than the model $\mathcal{M}$, and its input-output behavior approximates as well as possible the behavior of $\mathcal{M}$.

The emulator m can be either in an input-output or a state-space representation and one form may be more suitable than the other, depending upon the circumstances and the nature of the problem $\mathcal{P}$.

When an input-output representation is adopted, the emulator m is described entirely in the input-output space by a time-variant, generally non-linear transfer function.

$$\mathbf{y}_t = \mathbf{g}_t\left(\mathbf{y}_{t-1}, ..., \mathbf{y}_{t-p}, \mathbf{w}_{t-1}, ..., \mathbf{w}_{t-r}, \mathbf{u}_{t-1}, ..., \mathbf{u}_{t-s}, \mathbf{v}|\theta\right) \tag{1.3}$$

where θ is a parameter vector and p, r and s are suitable time-lags. On the other hand, when a state-space representation is considered, the emulator m is described by the following, more complex, state transition and output transformation functions:

$$\dot{\mathbf{x}}_{t+1} = \mathbf{f}_t \left(\mathbf{x}_t, \mathbf{w}_t, \mathbf{u}_t, \mathbf{v}|\theta\right) \tag{1.4a}$$

$$\mathbf{y}_{t+1} = \mathbf{h}_t \left(\mathbf{x}_t, \mathbf{w}_t, \mathbf{u}_t, \mathbf{v}|\theta\right) \tag{1.4b}$$

where $\mathbf{f}_t(\cdot)$ is a time-variant, generally non-linear vector function modeling the dynamics of $\mathbf{x}_t$, $\mathbf{h}_t(\cdot)$ is a a time-variant, generally non-linear, output transformation function, and θ is a vector of parameters.

1.3.2 Non-dynamic Emulation Modeling

When the problem $\mathcal{P}$ concerns the optimal planning of the functional $J(\cdot)$ with respect to the vector $\mathbf{v}$, or the uncertainty and sensitivity analysis of $J(\cdot)$ with respect to the parameters $\boldsymbol{\Theta}$, the emulation modeling effort on the identification of a static map between the planning variable $\mathbf{v}$ (and/or the parameters $\boldsymbol{\Theta}$) and the functional $J(\cdot)$.

Non-dynamic emulation modeling has been used extensively in a wide variety of mechanical and aerospace engineering studies (Queipo et al., 2005). Recently, it has been considered in the environmental modeling field, with applications in the planning of agroecosystems (Bouzaher et al., 1993; Børgesen et al., 2001; Krysanova and Haberlandt, 2002; Piñeros Garcet et al., 2006; Audsley et al., 2008), water distribution networks (Broad et al., 2005), groundwater resources (Rogers and Dowla, 1994; Aly and Peralta, 1999; Johnson and Rogers, 2000; Yan and Minsker, 2003; Kumar et al., 2010), and surface water resources (Solomatine and Torres, 1996; Neelakantan and Pundarikanthan, 2000; Bhattacharjya and Datta, 2005; Castelletti et al., 2010b,c; Sreekanth and Datta, 2010). In any case, non-dynamic emulation modeling can be considered as a simplified version of DEMo and, therefore, it is easily integrated within this wider concept and the subsequent discussion.

1.4 A general procedure for DEMo

The identification of a dynamic emulation model is made particularly difficult by the typically non-linear nature and large dimensionality of the model $\mathcal{M}$. A number of different approaches, and corresponding techniques, have been developed as the basis for finding ad-hoc solutions to specific problems. However, all of these approaches can be re-conducted to the following general categories:

1. In the *structure-based* approach, the mathematical structure of model $\mathcal{M}$ is 'manipulated', with the aim of deriving a simpler structure m. This approach is often adopted when the output $\mathbf{Y}_t$ of $\mathcal{M}$ is not defined, which is equivalent to saying that the output coincides with the state vector $\mathbf{X}_t$. Emulators identified using this approach are usually represented in a state-space form (eq. 1.4a).

2. *The data-based approach* identikits the emulator structure on the basis of a dataset $\mathcal{F}$ of state and output trajectories, obtained via simulation of the model $\mathcal{M}$ on a given horizon $\mathcal{H}$ under suitable input scenarios. The emulator structure can be either a black-box representation of some form; or a low order, conceptual, mechanistic model.

Whatever approach is adopted, the identification of an emulator can be structured as a six-step procedure (see figure 1.1). The first step (Step 1 - Design of experiments and simulation runs) concerns the generation of the data-set $\mathcal{F}$. This is obviously required for the data-based approach, but it is also necessary in the structure-based one for the evaluation of the emulator in Step 6. The variables (exogenous drivers and states) that will be operated by the emulator are obtained by aggregating, in some appropriate way, the variables in the model $\mathcal{M}$ and/or selecting, among them, the most relevant ones. These two, not necessarily mutually exclusive operations, are the core of the complexity reduction process performed by DEMo and are considered in two separate steps (Step 2 - *Variable aggregation* and Step 3 - *Variable selection*). Variable selection generally follows the aggregation because it can be more effectively performed on a reduced number of variables. Once these steps are complete, the emulator is eventually identified in Step 4 (Structure identification). Finally, in Step 5 - Evaluation and physical interpretation, the emulator is validated and a physical interpretation is provided. Note that, in any real application, many recursions through this procedure may be required. The details in each step of the emulation modeling procedure are described in the next section.

1.4.1 Design of Experiments and simulation runs

The Design Of computer Experiments (DOE), also known as Design and Analysis of Computer Experiments (DACE), is used to design a sequence of simulation runs

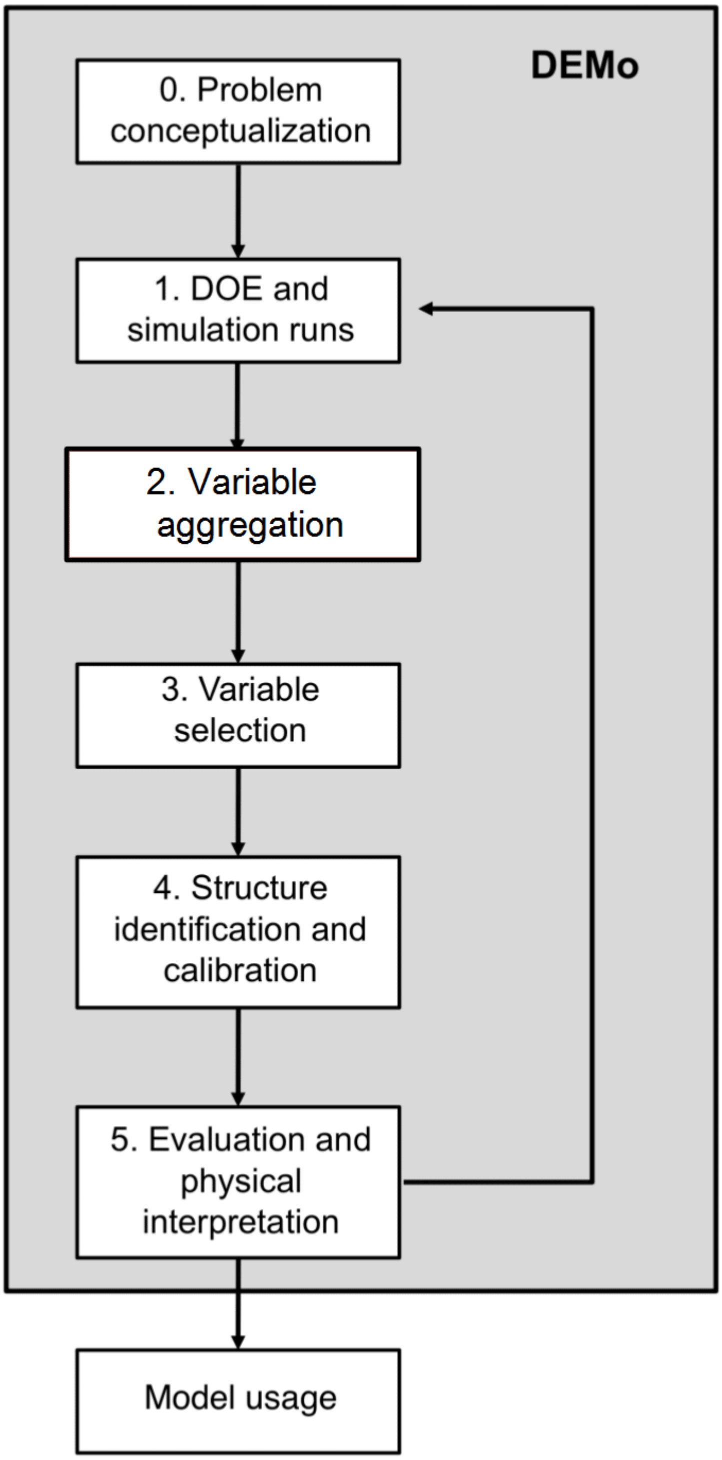

DEMo
0. Problem conceptualization
1. DOE and simulation runs
2. Variable aggregation
3. Variable selection
4. Structure identification and calibration
5. Evaluation and physical interpretation
Model usage

Figure 1.1: The DEMo procedure steps (see Castelletti et al., 2012b).

for the model $\mathcal{M}$ with the purpose of constructing the data-set $\mathcal{F}$ for the subsequent DEMo steps. This requires the specification of the input trajectories to the model $\mathcal{M}$ (i.e. the exogenous driver $\mathbf{W}_t$ and the control $\mathbf{u}_t$), as well as the values of the planning vector $\mathbf{v}$, that will drive the simulation runs, the parameters being set to their nominal value $\boldsymbol{\Theta}$.

In principle, the data-set $\mathcal{F}$ should be sufficiently informative, reproducing all the possible system behaviors and features, excited and forced by the spectrum of external forces, controls and planning variables that may occur given the problem $\mathcal{P}$. This can be ensured by relying on the procedures used in the design of dynamic experiments, such as those discussed in Goodwin and Payne (1977). In other words, the experiments have to be designed in such a way that all the dynamical modes of $\mathcal{M}$'s response that are of interest for $\mathcal{P}$ are activated.

However, according to the computational requirements for simulating $\mathcal{M}$ (i.e. the limit on the feasible number of simulation runs), a somewhat less formal experiment design may need be adopted (e.g. the historical observations available for the exogenous drivers and a well chosen periodic square wave input for the control, that allows the system to reach a steady state at each step). The accuracy requirements in the DOE also depends on the different approaches to the DEMo problem.

1.4.2 Variable aggregation

The purpose of this step is to aggregate the components of the state vector $\mathbf{X}_t$ (and of the exogenous driver vector $\mathbf{W}_t$) into lower dimensionality vectors. As common practice in environmental modeling, the model $\mathcal{M}$ is spatially-distributed: so the space discretization can lead to a strong increase in the dimensionality of the state and exogenous driver vectors.

The data generated via simulation in Step 1 (sometimes referred as snapshots) are used in an aggregation scheme to identify a mapping of the state $\mathbf{X}_t$ into a lower dimensional state $\tilde{\mathbf{X}}_t$, so that the majority of the variation in the $\mathbf{X}_t$ data is captured. The same is done with respect to $\mathbf{W}_t$, thus obtaining a reduced exogenous driver vector $\tilde{\mathbf{W}}_t$. The most simple and 'natural' aggregation scheme is based on the expert knowledge of the system (see Galelli et al., 2010; Castelletti et al., 2010a). This is particularly the case when $\mathcal{M}$ is spatially-distributed. Alternatively, formal and analytical aggregation techniques can be employed. Such techniques are commonly referred to as *feature extraction* techniques (Guyon et al., 2006). The linear technique for aggregation that has been adopted most often, up to now, is Principal Component Analysis (Jolliffe, 2005) (also known as proper orthogonal decomposition (Willcox and Peraire, 2002) or Karhunen Loeve Transform (Zhang and Michaelis, 2003), which performs a linear mapping of the data produced by the model $\mathcal{M}$ to a lower dimensional space in such a way that the variance of the data in the lower dimensional representation is maximized. The literature also presents a variety of non-linear feature extraction techniques (for a review, see Lee and Verleysen, 2007). Different approaches have been also suggested from the system and control literature in the last few decades, from simple methods, such as the Power Reduction approach of Liaw et al. (1986), to more complex procedures, such as Balanced Model Order Reduction, which are also used in the structure-based approach. They also include Dominant Mode Analysis (DMA) (see Young, 1999), which is used for model reduction in the DBM approach to emulation (see Young and Ratto, 2009).

1.4.3 Variable selection

Based on the information content of $\tilde{\mathbf{F}}$, model $\mathcal{M}$ is further simplified by selecting the components of $\tilde{\mathbf{X}}_t$ and $\tilde{\mathbf{W}}_t$ that will constitute the emulator's state $\mathbf{x}_t$ and exogenous driver $\mathbf{w}_t$ vectors. Generally, this operation relies on some automated technique, since $\tilde{\mathbf{X}}_t$ and $\tilde{\mathbf{W}}_t$ are often too large to be handled by a human operator.

1.4.4 Structure identification

The outcome of the variable selection (Step 3) are the variables characterizing the emulator, as well as the nature of the relationship between these variables and the output $\mathbf{y}_t$. This information can be exploited in this step of the DEMo procedure: in particular, this step is generally performed in two stages. The first stage is 'structure identification', and the second is 'parameter estimation': first, the structure of the function $\mathbf{g}_t(\cdot)$ (or $\mathbf{f}_t(\cdot)$ and $\mathbf{h}_t(\cdot)$) is identified (e.g. using model structure identification criteria), then the value of θ that characterizes the best model structure is estimated (optimally in some sense, if this is possible, but otherwise to yield statistically consistent estimates). In general, the emulator structure is only obtained tentatively in the first step, which serves as a 'screening' step for the variables to be finally included in the emulator. The class of functional relationships underlying the variable selection process (Step 3) is usually the first option for the structure identification (e.g. when correlation analysis is employed, a linear model is the most coherent choice) but, usually, the exploration of a wider class of models is more effective (Guyon and Elisseeff, 2003).

In any case, whatever approach is used, this step is concluded with a parameter estimation performed over the data-set $\tilde{\mathcal{F}}$ that provides the actual values for the θ parameters. If the performance measures are satisfactory, one can proceed with the following step; otherwise, one of the previous step must be re-considered.

1.4.5 Evaluation and physical interpretation

As introduced in Section 1.4.3, the emulator must be evaluated from two different points of view: i) it must reproduce as well as possible the input-output behavior of the model $\mathcal{M}$; ii) it must be credible. With respect to point i), the emulator is validated against that part of the data-set $\tilde{\mathcal{F}}$ that has not been used for the model identification (the validation data-set). As for point ii), the credibility of the emulator is directly related to its physical interpretability. This latter property is inherent when the emulator structure is obtained with the techniques proposed for the structure-based approach in Section 1.4.3; or with the data-based approach, when it can be satisfactorily interpreted in a physically meaningful manner. Generally, the identification of an emulator in state-space representation makes it easier to maintain a physically meaningful relationship between the emulator and the original model variables.

1.4.6 Model Usage

Once the emulator has been successfully validated against the data, it is ready to be employed by the user in the resolution of the problem $\mathbb{P}$. However, during the identification of the emulation model more than one run of the entire procedure can arise. In fact, if the performance of the model is not considered sufficient for the future use of the model itself, it's possible to design different simulation runs in order to evaluate other reduction approaches.

Preview of the remaining chapters

While Chapter 1 gives an overview of model reduction and introduces the general procedure of DEMo exercises, the remaining chapters apply that procedure to the case study, Tono Dam. Chapter 2 explains in detail the methodologies used in this study: RVS, PCA, and its extensions, namely, SPCA and WPCA. PCA techniques account for step 2 of the DEMo exercise, while RVS accounts for step 3 (See figure 1.1); they are the focus of this thesis. Step 1, the design of experiments and simulation runs was performed *a priori*, and its details are shown at the beginning of Chapter 4, after the case study is explained in Chapter 3.

Chapter 2

Methodology

2.1 Introduction

This chapter consists of the methodologies used in this thesis for model reduction. Taking into account the Dynamic Emulation Modeling (DEMo) procedure described in chapter 1 (See 1.1), the techniques used in this chapter, PCA and RVS, account for steps 2 and 3 of that procedure, respectively, which aim at reducing the number of variables in the model $\mathcal{M}$ to be used in the emulator $\mathbf{m}$. The techniques are divided into selection-based and projection-based model reduction techniques.

2.2 Recursive Variable Selection

2.2.1 Motivation

Representing selection-based model reduction is an automatic variable selection method introduced by Castelletti et al. (2011) and Galelli (2010) through which the variables that are most relevant to emulate the output of the process-based model $\mathcal{M}$ are selected. The selection is made such that the subset of selected variables can build an emulator which is both accurate and compact, i.e., it has an output close enough to the output of $\mathcal{M}$ while achieving a significant reduction in dimensionality. This is achieved through an automatic, data-driven method that recursively defines a sequence of variable selection problems, in which the accuracy is tuned to the desired emulator parsimoniousness.

2.2.2 Procedure

The RVS algorithm[1] (Castelletti et al., 2011) proceeds iteratively in three steps over each component of $\mathbf{Y}_t$. i) Given the information content of the dataset $\tilde{\mathcal{F}}$, the most relevant variables in explaining the given component are selected, with some appropriate *Input Selection* (IS) algorithm, among the components of the vectors $\tilde{\mathbf{X}}_t$, $\tilde{\mathbf{W}}_t$ and $\mathbf{u}_t$. This gives the arguments of the output transformation function (equation 1.4b) associated to the considered output. ii) For each state variable selected in the previous step, a new run of the IS algorithm is performed to select the variables relevant to describe its dynamics. This gives the arguments of the corresponding component of the vector state transition function (equation 1.4a) associated to the considered state variable. iii) If the second step leads to the selection of further variables from the vector $\tilde{\mathbf{X}}_t$ (i.e. state variables not yet included in $\mathbf{x}_t$), it is recursively repeated, until all the selected state variables are given a dynamic description. Once the RVS algorithm is over, the arguments of equations 1.4 are known.

Each invocation of the RVS algorithm requires to run an IS algorithm that selects the most relevant input variables to explain a specified output variable. Algorithms suitable for this task must account for both significance and redundancy: in other words, they must be able to select only the most relevant input variables, while trying to avoid the inclusion of redundant ones, which would unnecessarily add to the emulator complexity. They must also account for non-linearities. The following subsection presents the IS algorithm used in this study, the Iterative Input Selection algorithm (Castelletti et al., 2012a; and Galelli and Castelletti, 2013).

2.2.3 Iterative Input Selection

As mentioned previously, the ideal selection algorithm should account for non-linear dependencies and redundancy between variables, as real-world optimal management problems are usually characterized by non-linear dynamic models with multiple coupled variables. Moreover, it must be computationally efficient, since the number of candidate variables is generally large, particularly when the original process-based model is spatially distributed. To fulfill these requirements, Castelletti et al. (2012a) developed the Iterative Input Selection (IIS) algorithm[2], a model-free, forward-

[1] See Appendix A.3 for a pseudo code of the algorithm

[2] See Appendix A.4 for a pseudo code of the algorithm

selection algorithm.

Given the output variable to be explained and the set of candidate variables, the IIS algorithm first exploits an Input Ranking (IR) algorithm that provides the best performing input according to a global ranking based on a statistical measure of significance (preferably accounting for non-linear dependencies, as proposed by Wehenkel (1998)). To account for variable redundancy, only the most significant variable is then added to the set of selected variables. The reason behind this choice is that, once an input variable is selected, all the inputs that are highly correlated with it may become useless and the ranking needs to be re-evaluated. So, the algorithm proceeds first as follows: first it estimates, with an appropriate model building (MB) algorithm[3], an underlying model $\hat{m}(\cdot)$ to explain the output; then it repeats the ranking process using the residuals of model $\hat{m}(\cdot)$ as new output variable.

The algorithm iterates these operations until the best variable returned by the ranking algorithm is not in the already selected ones or the accuracy of $\hat{m}(\cdot)$ does not significantly improve. The accuracy can be computed with a suitable distance metric between the output and the model $\hat{m}(\cdot)$ prediction, or more sophisticated metrics accounting for both accuracy and parsimoniousness (e.g. the Akaike information criterion, Bayesian information criterion or Young identification criterion). In this thesis the accuracy of the model is expressed through the parameter R^2.

The choice of a suitable model building algorithm (MB) and ranking procedure (IR) is thus fundamental to let the IIS algorithm be capable of dealing with non-linearities, redundancy and high-dimension data-sets. Among the many alternative model classes, in this thesis Extremely randomized trees (or Extra-Trees, a tree-based method proposed by Geurts et al. (2006) that can provide all these desirable features) are used. As a consequence, also the choice of which ranking algorithm (Jong et al., 2004) to use has fallen on a method based on Extra-Trees, since their particular structure can be exploited too to infer the relative importance of the input variables. Finally, Extra-trees will also be used for step 4 of the DEMo exercise, i.e, structure identification and calibration, hence, the resulting Extra-tree models from the competing sets of reduced variables will be compared. In other words, Extra-tree models will be used as an evaluation tool, namely, for the accuracy of the reduction.

[3] Depending on whether a parametric or a non-parametric model structure is adopted for the underlying model, the model building (MB) algorithm can be either a traditional parameter estimate algorithm or the building algorithm of the regressor.

2.3 Principal Component Analysis

2.3.1 Motivation

This section describes model reduction through the procedure of Principal Component Analysis (PCA), also known as the Karhunen-Loève transform (Karhunen, 1947), Empirical Orthogonal Functions, or Proper Orthogonal Decomposition (POD). Given the information content of the dataset $\tilde{\mathcal{F}}$, namely, state variables $\tilde{\mathbf{X}}_t$, exogenous inputs $\tilde{\mathbf{W}}_t$, and control variables $\mathbf{u}_t$, PCA derives a new set of variables that are linear combinations of the original variables $\tilde{\mathbf{X}}_t$, $\tilde{\mathbf{W}}_t$, and $\mathbf{u}_t$. The transformation is defined in such a way that, generally, only a few of the new variables account for most of the variance of the original data-set. Therefore, dimesnionality reduction can be achieved by truncation of the new variable-set. This is the context in which PCA is used in this study.

2.3.2 Theory

Principal component analysis is a very old and popular multivariate statistical technique. The origin of PCA can be traced back to Pearson (1901), but the modern formalization was done by Hotelling (1933), who also coined the term *Principal Component*. PCA is used in many scientific disciplines to perform different tasks such as clustering, classification, and most notably, dimensionality reduction.

There are several ways to interpret what PCA does (Shlens, 2014), one of them is that is aims at finding a new linear basis to express the data which reveals the data structure better than the default (or naive) basis. In other words, it aims to measure the data from a new coordinate system about which the data is most spread-out (or variant). This interpretation explains the orthogonality of the transformation and the number of principal components being equal to the number of variables.

Formally, principal components are optimally-weighted linear combinations of the original variables. The weights are defined in such a way that gives the principal components the following properties: the first principal component explains the maximal amount of variance in $\tilde{\mathcal{F}}$, while the second principal component accounts for the maximal amount of variance in $\tilde{\mathcal{F}}$ that was not explained in the first principal component, and it is uncorrelated with the first component. Every remaining component has the same two properties: It accounts for the maximal amount of variance not explained by all preceding components, and it's uncorrelated with all of them. Therefore, principal components explain progressively less variance until the p-th principal component is reached.

2.3.3 Procedure

Given the defining properties of principal components in the previous section, the problem of finding the components is reduced to finding the appropriate weights (or coefficients), represented by p coefficient vectors (or *loading* vectors). Since the variance of the first components is to be optimized, the problem of finding the first loading vector can be be formulated as a constrained optimization problem.

Assuming the data of snapshots in $\tilde{\mathcal{F}}$ is standardized[4] and arranged in matrix $\mathbf{V}_{n\times p}$ where n is the number of observations, and p is the number of variables, and and assuming $\mathbf{W}_{n\times p}$ is the transformation matrix consisting of the loading vectors in the columns, which transforms $\mathbf{V}$ into a matrix of principal components, and $\mathbf{R}$ is the correlation matrix of the q variables in $\mathbf{V}$.

$$\mathbf{V} = \begin{pmatrix} v_{11} & \dots & v_{1p} \\ \vdots & \vdots & \vdots \\ v_{n1} & \dots & v_{np} \end{pmatrix} \tag{2.2}$$

The first principal component $\mathbf{PC}_1$ is defined by the following linear combination (Sadocchi, 1990):

$$\mathbf{PC}_1 = \mathbf{v}_1 a_{11} + \mathbf{v}_2 a_{21} + \ldots + \mathbf{v}_p a_{p1} = \mathbf{V}\mathbf{a}_1 \tag{2.3}$$

The variance of $\mathbf{PC}_1$ is equal to:

$$s^2_{\mathbf{PC}_1} = \sum_{i=1}^{p}\sum_{j=1}^{p}(a_{i1}a_{1j}\sigma_{ij}) = \mathbf{a}_1^T\mathbf{R}\mathbf{a}_1 \tag{2.4}$$

[4] Standardization is generally performed before applying the PCA procedure, specially when the variables are measured in different units (Baxter, 1995). In this context, a standardized variable v is computed as follows:

$$v = \frac{x-\mu}{\sigma} \tag{2.1}$$

where x is the original variable, μ is the mean of the population and σ is the standard deviation of the population.

In order to maximize the variance of $\mathbf{PC}_1$, the loading vector $\mathbf{a}_1$ is computed solving the following problem:

$$\mathbf{a}_1 = \arg\max_{\mathbf{a}_1}(\mathbf{a}_1^T\mathbf{R}\mathbf{a}_1) \tag{2.5a}$$

subject to

$$\mathbf{a}_1^T\mathbf{a}_1 = 1 \tag{2.5b}$$

The constraint (2.5b) normalizes $\mathbf{a}_1$ and moreover limits the values of the coefficients a_{i1} preventing the construction of a principal component $\mathbf{PC}_1$ with a variance infinitely high.
The numerical value of $\mathbf{a}_1$ is obtained using the method of Lagrange multipliers:

$$\frac{\partial}{\partial\mathbf{a}_1}[s^2_{\mathbf{PC}_1} + \lambda_1(\boldsymbol{\lambda} - \mathbf{a}_1^T\mathbf{a}_1)] = \frac{\partial}{\partial\mathbf{a}_1}[\mathbf{a}_1^T\mathbf{R}\mathbf{a}_1 + \lambda_1(\boldsymbol{\lambda} - \mathbf{a}_1^T\mathbf{a}_1)] = 2(\mathbf{R} - \lambda_1\mathbf{I})\mathbf{a}_1 \tag{2.6}$$

Thus the coefficients a_{i1} are the solution of the following system of p linear equations:

$$(\mathbf{R} - \lambda_1\mathbf{I})\mathbf{a}_1 = 0 \tag{2.7}$$

where λ_1 is the solution of the characteristic equation:

$$|\mathbf{R} - \lambda_1\mathbf{I}| = 0 \tag{2.8}$$

Therefore λ_1 is an eigenvalue of the correlation matrix $\mathbf{R}$ and $\mathbf{a}_1$ is the corresponding eigenvector. Moreover, λ_1 is the maximum eigenvalue, because it is the solution of system (2.7) that, multiplied by $\mathbf{a}_1^T$, results:

$$\lambda_1 = \mathbf{a}_1^T\mathbf{R}\mathbf{a}_1 = s^2_{\mathbf{PC}_1} \tag{2.9}$$

Then it is possible to define the second principal component $\mathbf{PC}_2$:

$$\mathbf{PC}_2 = \mathbf{v}_1 a_{12} + \mathbf{v}_2 a_{22} + \ldots + \mathbf{v}_p a_{p2} = \mathbf{V}\mathbf{a}_2 \tag{2.10}$$

The coefficients a_{i2} have to compelled to the constraint $\mathbf{a}_2^T\mathbf{a}_2 = 1$ to maximize the variance of $\mathbf{PC}_2$ and the constraint $\mathbf{a}_1^T\mathbf{a}_2 = 0$ in order to have orthonormal and independent components. So the total explained variance of the first n components is equal to the sum of the variances of each component. Applying again the method of Lagrange multipliers, the coefficients a_{i2} are the solution of the following system:

$$(\mathbf{R} - \lambda_2\mathbf{I})\mathbf{a}_2 = 0 \tag{2.11}$$

So, the coefficients a_{i2} are the element of the eigenvector associated to the second largest eigenvalue of the correlation matrix $\mathbf{R}$.
In general it is possible to define the j-the principal component as:

$$\mathbf{PC}_j = \mathbf{v}_1 a_{1j} + \mathbf{v}_2 a_{2j} + \ldots + \mathbf{v}_q a_{qj} = \mathbf{V}\mathbf{a}_j \tag{2.12}$$

where the coefficients are the elements of the eigenvector of the correlation matrix $\mathbf{R}$ associated to the j-th largest eigenvalues λ_j. The variance of $\mathbf{PC}_j$ is equal to λ_j and therefore the total variance of the whole system is $\lambda_1 + \ldots + \lambda_q = tr\mathbf{R} = p$. The importance of the j-th principal component is thus λ_j/q (Morrison, 1976).

Finding the loading vectors by computing the eigenvectors of the correlation matrix[5] is the first analytical solution to PCA problem. It is a one-shot solution, as it is able to compute all components at once, and not sequentially. Another popular and equivalent analytical soultion to PCA uses the Singular Value Decomposition or SVD, and is performed as follows:

Let the SVD of $\mathbf{V}$ be:

$$\mathbf{V} = \mathbf{U}\mathbf{D}\mathbf{B}^T \tag{2.13}$$

$\mathbf{Z} = \mathbf{U}\mathbf{D}$ are the principal components of $\mathbf{F}$, and the columns of B are the corresponding loading vectors. The sample variance of the j-th principal component is D_{jj}^2/n.

2.3.4 Analysis

Which principal components to retain?

After finding the principal components, it is important to appropriately determine how many and which principal components to retain. Since principal components explain progressively less variance as their order increases, it is commonplace to retain the first q components ($q < p$). The number can be limited by different criteria, alot of which are based on explained variance.

1. *Threshold on cumulative variance*: Only the components containing a predefined percentage of the total variance (e.g. 75% or 90%) are retained.

[5] Equivalently, the same procedure can be performed using the *covariance* matrix. It is important to standardize the data when using this approach.

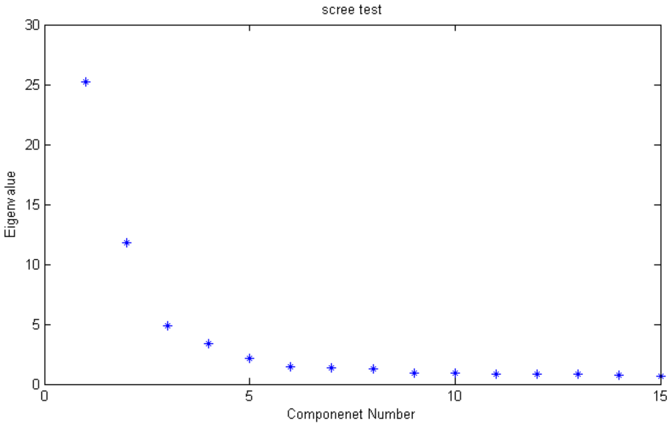


Figure 2.1: Scree plot

2. *Eigenvalue one criterion*: Also known as the Kaiser criterion (Kaiser, 1960), which states that only the components with a corresponding eigenvalue λ_j larger than 1 are retained. Any component satisfying this criterion explains more than one p-th of total variance. More generally, one could only retain components that explain a pre-defined percentage of total variance (e.g. 1% or 5%).

3. *Threshold on the variance of the next component*: Stopping when the percentage explained variance of the next component is less than a pre-defined threshold (e.g. 1% or 5%).

4. *Scree test* (Cattell): In this test, the components are plotted with their corresponding eigenvalues (Figure 2.1), and if a *break* or a large drop of the eigenvalues is found at one component, only components before the break are retained.

The above criteria are based on explained variance, and always result in selecting the *first* q components. They are often effective for retaining the most meaningful components for many application if the threshold is set carefully. However, large explained variance does not necessarily guarantee an accurate emulator. In some cases, components that explain less variance may be more relevant in explaining the dynamics of the emulator's output. In those cases, *variable-selection* techniques can be used to select the most relevant components regardless of their order or explained variance. Other ways to select principal components that are not-based on explained variance can be found in Mei et al. (2008).

Adequate sample size for PCA

Having an adequate sample size is an important factor to consider before applying PCA to a data-set. "Larger samples are better than smaller samples (all other things being equal) because larger samples tend to minimize the probability of errors, maximize the accuracy of population estimates, and increase the generalizability of the results" (Osborne and Costello, 2004).

Generally, similar guidelines to those given for multiple regression problems can be followed for PCA, namely, guidelines concerning the ratio between variables and observations, e.g. 1:30 or 1:50 are suggested in (Pedhazur, 1997). Another approach is giving suggestions on the minimum adequate sample size rather than the variable to observation ratio.

Moreover, the effectiveness of the sample size for PCA can be assessed empirically. Kocovsky et al., 2009 analytically measure the variation in the sign and magnitude of the first principal component loadings as a function of the sample to variable ratio, as well as the correlation among the principal component loadings from different resamples of the same size.

Some consideration on sample size may also be application-dependent. For example, in this study, the sample size must be suitable for both PCA and RVS; the technique to which is it compared. Moreover, since the reduction aims at creating an emulator to be used for optimal planning, the sample size must account for that as well, i.e., the sample size must be small enough to make the optimal planning computationally efficient.

One of the major drawbacks of PCA is that each principal component, in general, is a combination of all of the original variables, i.e., all loadings are typically non-zero. This makes it difficult to interpret the PCs. Sparse Principal Component Analysis (SPCA) was developed to solve that problem (Zou et al., 2004), and it is introduced in the section.

2.4 Sparse Principal Component Analysis

The goal of Sparse Principal Component Analysis is to introduce sparsity in the principal componenets, i.e., to reduce the number of variables that load on the PCs by having loading vectors with a limited number of non-zero loadings. This desired effect can be achieved following one of several approaches:

- Performing standard PCA and then artificially setting to zero loadings with an absolute value beneath a certain threshold. This is an informal, yet widely used in practice technique, but it can be misleading (Cadima and Jolliffe, 1995).

- Restriction of coefficients to take values from a small set of allowable integers, such as $0, 1, -1$ (Vines, 2000): a series of linear transformations maximize, in each transformation, the variance of the data with respect to one of the transformed axes. The transformation is restricted so that transformed axes can still be represented by simple directions, defined as a direction that can be represented by a vector proportional to an integer vector. Furthermore the transformed axis for which the data have the greater variance will tend to be proportional to a vector of small integers and hence be particularly simple.

- Reformulating the standard PCA cost function in a way that naturally induces sparsity.

The last approach is the formal one, and it is the one considered in this thesis. This is done by modifying the optimization problem with either a constraint or penalty that cause sparsity in the resulting loading vectors.

Zou et al. (2006) introduced a formulation of the problem called the *elastic net*, which is a combination of *Ridge Regression* and LASSO (least absolute shrinkage and selection operator) regression (Tibshirani, 1996). LASSO refers to imposing an L_1-norm penalty in the PCA cost function. A generic lasso regression problem can be defined as follows:

1. Let $Y = (y_1, \ldots, y_n)^T$ be the response vector and $\mathbf{X} = (X_1, \ldots, X_n)^T, j = 1, \ldots, p$ the predictors, where $X_j = (x_{1j}, \ldots, x_{nj})^T$, $\hat{\beta}_{lasso}$ is the lasso estimate obtained by minimizing the lasso cost function, and λ is a non-negative parameter that controls the sparsity.

$$\hat{\beta}_{lasso} = \underset{\beta}{argmin} \|Y - \sum_{j=1}^{p} X_j \beta_j\|^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{2.14}$$

The nature of the L_1-norm[6] continuously shrinks the coefficients toward zero while simultaneously achieving good prediction accuracy. If λ is high enough, some coefficients will be shrunk exactly to zero. This produces accurate and sparse models (Zou et al., 2006).

2. The second part of elastic net, the ridge regression, extends the problem by adding an L_2-norm penalty. This extension aims at overcoming the limitation that lasso has when dealing with data-sets that have a low number of observation and a high number of variables, e.g., microarray data. This limitation was pointed out by (Zou and Hastie, 2005). For any non-negative λ_1 and λ_2, the elastic net estimate $\hat{\beta}_{en}$ is given as follows:

$$\hat{\beta}_{en} = (1 + \lambda_2) \left\{ \underset{\beta}{argmin} \|Y - \sum_{j=1}^{p} X_j \beta_j\|^2 + \lambda_2 \sum_{j=1}^{p} |\beta_j|^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| \right\} \tag{2.16}$$

The PCA solution is directly connected to the regression method in equation 2.16. The sparse loadings can be found as follows:
Let $\mathbf{X}$ be the data-matrix $(n \times p)$, and Z_i the i-th principal component, the sparse loadings estimate can be found as follows:

$$\hat{\beta}_{en} = \underset{\beta}{argmin} \|Z_i - \mathbf{X}\beta\|^2 + \lambda |\beta\|^2 + \lambda_1 \|\beta\|_1, \tag{2.17}$$

where $\|\beta\|_1$ is the 1-norm of β, λ and λ_1 are non-negative. The i-th estimating principal components can be found as $\mathbf{X}\hat{V}_i$ where $\hat{V}_i = \frac{\hat{\beta}}{\|\hat{\beta}\|}$. Equation 2.17 is referred to as *naive elastic net* by Zou and Hastie (2005), since it differs from elastic net (Eq. 2.16) by the scaling factor $(1 + \lambda)$, as the scaling does not affect $\hat{V}_i$, which is normalized.

The formulation in Eq. 2.17 depends on knowing Z_i, the result of the standard PCA to find the sparse approximation.

Alternatively, a self-contained regression formulation was introduced by (Zou et al., 2006) which finds the fist k sparse loading vector approximations without having to find the standard principal components first.

[6] The L_1-norm of a vector $\mathbf{x}$ is defined as the sum of the absolute values of its elements.

$$\|\mathbf{x}\| = \sum_{i=1}^{n} |x_i| \tag{2.15}$$

Let $\mathbf{A}_{p\times k} = [\alpha_1, \dots, \alpha_k]$ and $\mathbf{B}_{p\times k} = [\beta_1, \dots, \beta_k]$, then:

$$\left(\hat{\mathbf{A}}, \hat{\mathbf{B}}\right) = \underset{A,B}{argmin} \sum_{i=a}^{n} \|x_i - \mathbf{A}\mathbf{B}^T x_i\|^2 + \lambda \sum_{j=1}^{k} \|\beta_j\|^2 + \sum_{j=1}^{k} \lambda_{1,i} \|\beta_j\|_1 \tag{2.18a}$$

subject to

$$\mathbf{A}^T\mathbf{A} = I_{k\times k} \tag{2.18b}$$

where $\lambda_{1,j}$ $j = 1 : k$ is the sparsity controlling parameter, which can be set differently for each principal component, while λ is the same for all k components, and it is required to be positive only if $p > n$.

In addition to equation 2.18, other optimization formulations can used to produce sparse loading vectors. Richtárik et al. (2012) suggest 8 different formulations by combining the following: two norms for measuring variance (L_2, L_1), and two norms for inducing sparsity (L_0, L_1), which are used in two ways (constraint, penalty).

Table 2.1: 8 SPCA formulations (Richtárik et al., 2012)

#	Variance	SI norm	SI norm usage	X	$f(x)$
1	L_2	L_0	constraint	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1, \|x\|_0 \leq s\}$	$\|Ax\|_2$
2	L_1	L_0	constraint	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1, \|x\|_0 \leq s\}$	$\|Ax\|_1$
3	L_2	L_1	constraint	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1, \|x\|_1 \leq \sqrt{s}\}$	$\|Ax\|_2$
4	L_1	L_1	constraint	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1, \|x\|_1 \leq \sqrt{s}\}$	$\|Ax\|_1$
5	L_2	L_0	penalty	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1\}$	$\|Ax\|_2{}^2 - \lambda\|x\|_0$
6	L_1	L_0	penalty	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1\}$	$\|Ax\|_1{}^2 - \lambda\|x\|_0$
7	L_2	L_1	penalty	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1\}$	$\|Ax\|_2 - \lambda\|x\|_1$
8	L_1	L_1	penalty	$\{x \in \mathbf{R}^p : \|x\|_2 \leq 1\}$	$\|Ax\|_1 - \lambda\|x\|_1$

Table 2.1 shows the 8 SPCA formulations classified by: norm type, sparsity-inducing norm-type, and the usage of the sparsity-inducing norm. The L_2 variance formulations, numbers (1, 3, 5, and 7), were previously studied in literature. For instance, the formulation in equation 2.18 roughly falls under formulation 7 in the table; L_2 norm, L_1 penalized. On the other hand, L_1 variance formulations are less popular and were more recently proposed in literature (Meng et al., 2012; and Richtárik et al., 2012).

2.4.1 Matrix deflation

Deflation is a process that modifies a matrix to eliminate the influence of one of its eigenvectors. In the context of SPCA, deflation is necessary in cases where the problem is formulated such that it can only be solved sequentially, i.e., by solving the cost function to find the first loading vector, and then solving another cost function to find the next loading vector, and so on.

Typically, after one leading vector x_t is computed, the covariance matrix A_t is *delfated* with x_t resulting in a deflated covariance matrix A_{t+1}. If the same cost function is then solved on A_{t+1} instead of A_t, the resulting loading vector x_{t+1} will account for the maximal amount of variance not explained by x_t.

There are several methods to perform the deflation process (See Mackey (2009) for details). The most popular method, and the one used in this thesis is Hotelling's deflation, which is computed as follows:

$$A_{t+1} = A_t - x_t x_t^T A_t x_t x_t^T \tag{2.19}$$

2.4.2 Adjusted explained variance

The standard principal components are uncorrelated and their loading vectors are orthogonal. Therefore, the explained variance of the j-th principal component can be computed as λ_j/q, where λ_j is the corresponding eigenvalue and q the number of variables. On the other hand, SPCA guarantees neither the orthogonality nor the uncorrelated property. Therefore, when computing the total explained variance of k sparse principal componenets, the correlation between them must be accounted for. For instance, the total explained variance of the first $k+1$ sparse PCs should equal the explained variance of the first k PCs plus the explained variance of the extra $(k+1)-th$ PC, minus the variance attributed to new PC's correlation with the first k PCs.

Zou et al. (2006) propose a formula to compute the total explained variance of a group of sparse PCs which takes into account the correlation among them. Let $\hat{Z}$ be a group of sparse PCs computed by any method, and $\hat{Z}_{j.1,\dots,j-1}$ the residual after removing the linear dependence between $\hat{Z}_j$ and $\hat{Z}_1, \dots, \hat{Z}_{j-1}$:

$$\hat{Z}_{j.1,\dots,j-1} = \hat{Z}_j - \mathbf{H}_{1,\dots,j-1}\hat{Z}_j, \tag{2.20}$$

where $\mathbf{H}_{1,\ldots,j-1}$ is the projection matrix on $\{\hat{Z}_i\}_1^{j-1}$. Then the adjusted variance of $\hat{Z}_j$ is $\|\hat{Z}_{j.1,\ldots,j-1}\|^2$. This can be computed using the QR decomposition[7] $\hat{Z} = QR$:

$$\|\hat{Z}_{j.1,\ldots,j-1}\|^2 = R_{jj}^2 \tag{2.21}$$

and the total explained variance for k PCs is equal to $\sum_{j=1}^{k} R_{jj}^2$.

2.4.3 Numerical solutions

Two approaches to obtain sparse principal components are used in this thesis: The elastic net method by Zou et al. (2006), and the first formulation of table 2.1 by Richtárik et al. (2012). The former is an L_1 penalty problem, while the latter is an L_0 constraint problem, making them distinct approaches to induce sparsity. Moreover, each method was solved using different algorithms. Zou et al. (2006) proposes an alternating algorithm to solve the non-convex problem in equation 2.18 (See Zou et al., 2006 for details). The second approach is reformulated by Richtárik et al. (2012) to be solved by a generic *alternating maximization* method, which finds a closed form solution to all formulations in table 2.1. For software implementation, a Matlab toolbox developed by Sjöstrand et al. (2012) was used to solve the first approach, while for the second approach, a Matlab toolbox[8] was developed within the scope of this thesis.

2.5 Weighted Principal Component Analysis

In most implementations of ordinary PCA, all variables and all observations (samples) are given the same importance by pre-standardization of the data, and performing PCA by finding the eigenvectors of the convariance matrix. This allows variables of different units and scales to be comparable, but at the same time, it makes PCA inappropriate when a priori information is available about the variables or the samples, i.e., their relative importance; or when the data is contaminated with noise. Weighted Principal Component Analysis or WPCA is a modification to ordinary PCA that aims to address these cases.

[7] The QR decomposition of a matrix Z is defined as $Z = QR$ where Q is orthonormal and R is upper triangular

[8] `https://github.com/amjams/spca_am.git`

WPCA does not have a standard form, and usually consists of ad-hoc weighting schemes of either the samples or the variables. Examples of sample-wise weighting schemes can be found in Cheng et al., 2011, where they use WPCA to take into account the relative importance of pixels (samples) for identifying the locations of intrusive bodies from geochemical data. Pinto da Costa et al. (2011) apply the same principal to microarray data by defining a new correlation coefficient that gives higher weights to observations that are considered to be more important. Conversely, weighting schemes can be variable-wise, Yue and Tomoyasu (2004) propose an ad-hoc variable-wise weighting scheme to incorporate sensor knowledge in a fault detection problem.

For the Tono case study, a variable-wise weighting scheme is appropriate. The a priori knowledge that can be introduced to the problem relates to the fact that the principal components will be used to build emulators with different outputs (g^{temp}, g^{sed}, and g^{algae}). The relative importance of the input variables to any of the outputs is incorporated into PCs using a weighting scheme that modifies the covariance matrix.

$$cov(V) = (V^T V)/(N-1) \tag{2.22a}$$

$$cov_w(V) = (WV^T XV^T)/(N-1) \tag{2.22b}$$

where N is the number of samples, V is the standardized data matrix, $cov(V)$ is the covariance matrix, $cov_w(V)$ is the weighted covariance matrix, and W is an appropriate weight matrix.

Three weight matrices are suggested. Two of them incorporate the information from the IIS ranking[9], while the third uses *Pearson's* linear correlation coefficient. They are defined as follows:

- W_{IIS1}: Linearly spaced positive scalar values in the range $(1:100)$ are assigned to the IIS ranking order of the input variables for each emulator output, i.e., the input variable that ranks first in IIS is given a weight of 100, while the input variables ranked last is given a weight of 1, while other selected variables are weighted between 1 and 100 on a linear scale according to their ranking.

[9] The ranking is taken from 10 runs of the IIS algorithms on each output, and the variables were ranked according to the number of times they are selected, and their average relative δR^2 contribution

Variables that are not selected at all by IIS are given a weight of 0.5, i.e., they are weighted down, or given less importance than in the ordinary PCA case. Finally, the weight matrix W_{IIS1} is a diagonal matrix, with each diagonal element w_{jj} set to equal the weight associated to the j-th variable.

- W_{IIS2}: Defined similarly to W_{IIS1}, but with a logarithmic scale between $(1 : 100)$ instead of a linear scale. Variables that are not selected are weighted down by 0.1.
- $W_{pearson}$: In this scheme, the weights, i.e. the diagonal elements of $W_{pearson}$, are set as the absolute values of the Pearson's correlation coefficients between the variables and the output of interest.

This results in 9 weight matrices, and 9 sets of weighted principal components, three for each of the emulator outputs. In chapter 4, the accuracy of the models built with these PCs is compared to those built with ordinary PCs and sparse PCs.

Chapter 3

The Case Study: Tono Dam

This chapter starts with a description of the Tono Dam system which is used as the case study in this thesis. Then, a description of DYRESM-CAEDYM, the $1D$ spatially-distributed model used to describe the in-reservoir hydrological and ecological processes is presented. Finally, the optimal management problem of the Tono Dam is formulated, including a description of the outputs that are used for the emulation.

3.1 Description of the case study

3.1.1 System description

Tono dam is an artificial reservoir in western Japan constructed at the confluence of Kango and Fukuro rivers (see figure 3.1). It has a height of 75 m and it forms an impounded reservoir of 12.4×10^6 m^3 (gross capacity), with a surface area of 0.64 km^2 and fed by a 38.1 km^2 catchment.

The reservoir has been built for multiple purposes: it provides water for irrigation to several agricultural districts downstream (for a total irrigated surface area of 353 ha), feeds a hydro-power station with 1.1 MW installed capacity, provides industrial water supply of 30×10^3 m^3/day and drinking water supply of 20×10^3 m^3/day to the local communities, is used for buffering river floods (up to 5.5×10^6 m^3), provides ecological services (e.g. fish habitat) in the downstream river, and finally it is used for recreation.

The dam is equipped with a withdrawal intake tower for releasing active storage water at different levels. The selective withdrawal structure (SWS) is equipped with

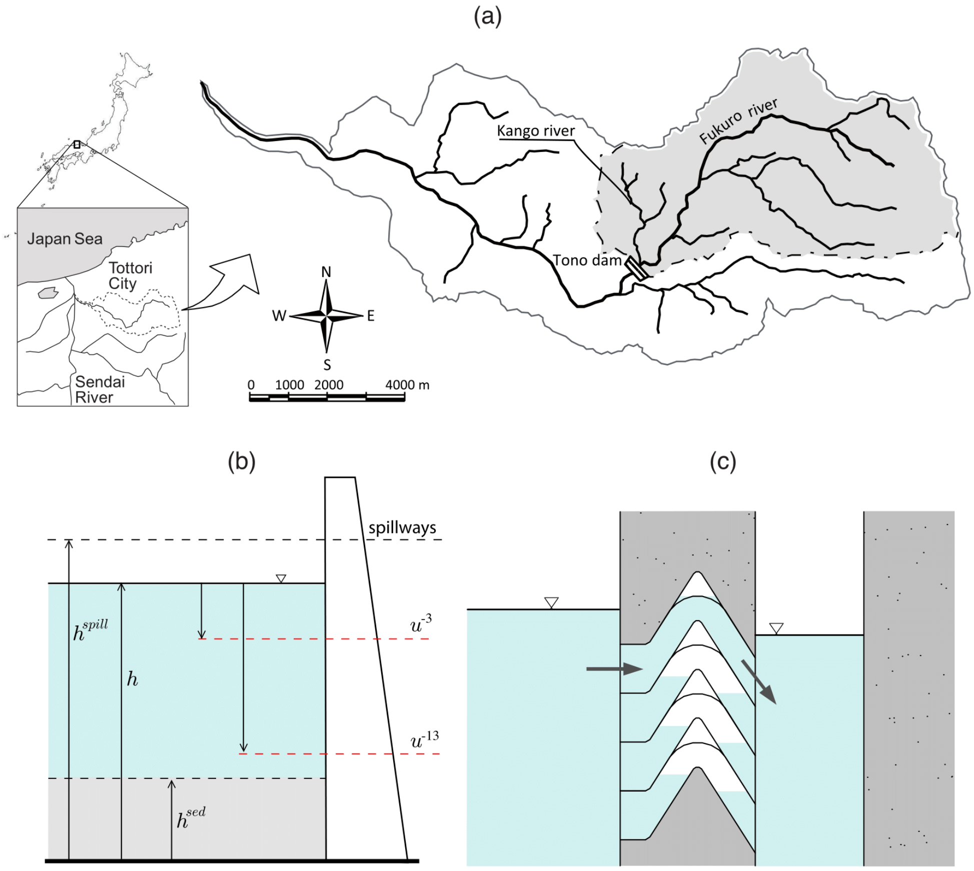


Figure 3.1: Tono Dam location in Western Japan (panel a), the main characteristics of the reservoir with two of the decision variables adopted in this study (panel b), and the scheme of the selective withdrawal structure (SWS) (panel c).

a rack of 15 vertically stacked siphons (see figure 3.1), starting at 18 m from the reservoir bottom. Siphons are operated by inflating or deflating air, and blending is allowed. The total amount of water released through the SWS is equally divided among the open siphons and it is conveyed into the hydro-power plant. A bypass is available just before the plant to divert any flow rate smaller or greater than those conveyable into the plant (1.0 m^3/s and 3.0 m^3/s respectively).
A flood orifice gate is foreseen at the elevation of 182.8 m (37.8 m from the bottom) just at the bottom of the flood buffering layer. Selective release is not available in the sediment storage, however an intake has been planned at 156 m to cover the Minimum Environmental Flow and supporting the outflow in exceptionally dry years, when the water level drops below the lower bound of the active storage. In normal conditions, the minimum environmental flow is guaranteed through the siphons. When the level drops below the SWS lower limit, the sediment outlet is activated. This intake can not be used for flushing away sediments.

While one of the main purposes of Tono dam operation is to provide water for irrigation, the SWS might have an impact on several other water uses. We distinguish between *in-reservoir* and *downstream* issues, the former being affected by level variations, the latter by the release.

In-reservoir

- ***Level*** Too low reservoir *levels*, which can be generated in the attempt to release water to satisfy agricultural water demand, can potentially reduce the recreational value of the lake. In order to emphasize this recreational interest, the SWS management has to consider to keep the lake level as close as possible to a reference level of 182.8 m a.s.l. as the normal high water level. This, however, implies stocking a significant volume of water in the reservoir with potentially negative effects both in-reservoir, e.g., boosting algal blooms, and downstream, e.g., water shortages.

- ***Algal blooms*** Odors and unattractive appearance of *algal blooms* can detract from the recreational value of the lake affecting the quality of the water stored in the reservoir. The physical processes driving the bloom of algae are particularly complex. However, thermal stratification has a dominant role. Controlling the temperature profile is a mechanical way of controlling the depth of nutrient load intrusion and therefore the algae bloom, which is basically sensitive to the available light. Moreover the temperature profile might vary as a consequence of withdrawing at different levels (Gelda and Effler, 2007). Generally, the deeper the withdrawal the more the deepening of the thermocline. Yet, this implies releasing colder water with potentially negative effects downstream and might affect sedimentation in the way explained below.

- ***Sedimentation*** High levels of in-reservoir *sedimentation* can remarkably reduce the reservoir life by inducing the rapid silting of the impoundment. Sedimentation is basically driven by the inflow and re-suspension can be assumed as negligible considering the reservoir depth (Evans, 1994). In particular, inflow intrusion is governed by the in-reservoir temperature profile and the inflow temperature because floods are more likely to intrude just above the thermocline (Yajima et al., 2006). Therefore, to maximize sediment evacuation, the release should be set at the depth at which the turbid inflow is intruding and then, if necessary, dynamically moved to the deeper siphons to intercept the maximum concentration of suspended solids not yet evacuated. Moreover, some recent studies (Yajima et al., 2006) have shown that using the top siphon combined with the spillways leads the inflow to the shallower depth and facilitate sediment flushing from the spillway. These ways of operating the SWS might have negative effects on the other sectors, like, for instance, the ecosystem downstream, which might be damaged by too warm water. Also recreation could be affected, since by keeping the thermocline in the shallow layer, algal blooms are more likely to occur as explained above.

Downstream

Irrigation and *Temperature* are the sectors identified downstream from the dam.

- ***Irrigation*** Farmers are interested in reducing the water supply deficit, which has a direct effect on the seasonal harvest and, therefore, on the annual income, which is the criterion through which the farmers judge the level of attractiveness of an operating policy (Hashimoto et al., 1982).

- ***Temperature*** The riverine ecosystem downstream from the dam is potentially threatened by large deviations of the water temperature from the seasonal natural patterns that might negatively affect faunal richness in both fishes and invertebrates (Bartholow et al. (2001) and references therein). According to

Fontane et al. (1981) and Baltar and Fontane (2008), a simple and physically rooted criterion to reduce the effect of artificially induced temperature variations is to force the outflow temperature to be as closest as possible to the (natural) inflow temperature.

3.1.2 Evaluation Criteria

In principle, each one of the sector criteria specified in Section 3.1.1 has to be associated to one or more quantitative evaluation criteria, through which different control policies can be evaluated and compared.

The evaluation criteria are defined as the aggregation over a pre-selected evaluation time horizon H^{val} (1990-1994) of step-costs g_{t+1}, as follows:

$$z = \frac{1}{H^{val}} \sum_{t=1}^{H^{val}-1} g_{t+1} \tag{3.1}$$

Level

The first step-cost is associated to water-level, and it is defined as the squared positive difference of lake level with respect to the reference level $\bar{h} = 182.8\,m$.

$$g_{t+1}^{lev} = \left[max\left(\bar{h} - h_{t+1}, 0\right)\right]^2 \tag{3.2}$$

Algal bloom

The step-cost associated to the recreation sector of interest has to be strongly correlated to algal bloom. One suitable choice would be to consider phosphorous concentration in the epilimnion, but this is hardly measurable[1]. The average concentration of Chl-a in the epilimnion is a potential valuable alternative. However, since Chl-a is acting as a proxy of algal bloom, the daily average hourly maximum concentration of Chl-a in the see-through layer was considered as follows:

$$g_{t+1}^{rec} = \frac{1}{24} \sum_{\tau=1}^{24} \max_{z_\tau \in z_E} (chla_\tau(z_\tau))^\alpha \tag{3.3}$$

[1] Another potential reference variable is the average value of DO (Dissolved Oxygen) in the reservoir bottom layer. Maximizing the value of the oxygen dissolved in the water reduces the chance for anoxic conditions in the bottom layer with consequent release of nitrogen and increase in algal bloom via water mixing. However, this only accounts for the nutrient load available in the bottom layers of the reservoir, but ignores the nutrient contribution coming in with the inflow.

where $chla_\tau$ is the Chlorophyll a concentration [g/m^3] at the τ-th hour of day t, z_τ is the depth with respect to the lake surface, z_E is the see-through layer depth (E stands for euphotic layer), set at 7 metres below water surface, as it came out from the analysis on the transparency features of the water, and α is an amplifying coefficient to take into account the associated bloom effect (provisionally $\alpha = 1$).

Sedimentation

The step-cost associated to this sector is the daily volume of sediment expelled with the release, which has to be maximized in order to reduce the silting of the reservoir and increase its expected life. Therefore, the considered step-cost is the following:

$$g_{t+1}^{sed} = TSS_{t+1}^{out} \tag{3.4}$$

where TSS_{t+1}^{out} is the amount of Total Suspended Solid [g/day] in the reservoir outflow between t and $t+1$. More precisely, TSS_{t+1}^{out} can be computed as

$$TSS_{t+1}^{out} = \sum_{i=1}^{N} tss_{t+1}^{i} r_{t+1}^{i} + tss_{t+1}^{spill} r_{t+1}^{spill} \tag{3.5}$$

where r_{t+1}^i [m^3/day] is the volume of water actually released[2] from the i-th siphon of the N available in the SWS and tss_{t+1}^i is the average TSS concentration [g/m^3] in the corresponding layer, tss_{t+1}^{spill} is the average TSS in the layer of the spillway, and r_{t+1}^{spill} the actual release from the corresponding layer.

Irrigation

The step-cost associated to this sector is the water daily deficit. Since the impact of deficit on the plant growth might have different effects depending on the vegetation

[2] A fundamental modeling tool in designing daily control policies is the so-called ***release function*** (see Soncini-Sessa et al. (2007a) for more details). Indeed, the release decision $\mathbf{u}_t$ is taken at time t and is supposed to be implemented between t and $t+1$, when a new decision is taken. Since the reservoir level might change as a consequence of the inflow, of the evaporation and of the water being released, the actual volume r_{t+1} released from the SWS at the end of the interval $[t, t+1)$ might not correspond to the release decision taken at time t. For example, due to high inflow, the water level can rise above the spillways bottom level, and some water can flow out uncontrolled from the spillway orifice. Another example, the water being release can cause the level to drop below the active siphon and the release decision has to be allocated to other siphons.

phase and the effect of the water deficit on the real crop stress is not linear (reflecting some risk aversion by the farmers), the following step-cost was defined:

$$g_{t+1}^{irr1} = \beta(t)\left(\left(w_t - (r_{t+1} - q_{t+1}^{MEF})\right)^+\right)^\gamma \tag{3.6}$$

where w_t is the agricultural reshaped (as explained later) water demand, r_{t+1} is the total actual release from the dam (including SWS and spillway), q_{t+1}^{MEF} is the Minimum Environmental Flow and $(\cdot)^+$ is a mathematical operator returning only positive values of the deficit or zero. $\beta(t)$ is a time-varying coefficient taking into consideration the different relevance of water deficit in different periods of the years and γ is a parameter accounting for the risk aversion of the farmers and is to be selected into a range of 1-12 (see Soncini-Sessa et al. (2007a)).

The nominal water demand provided by the Ministry refers to a section downstream of a lateral tributary to the downstream main river, whose flow is evaluated via simple regression to be 48% of the inflow to Tono dam[3]. Not taking into account this other source of water would assign to the farmers more water than their needs. It is therefore necessary to reshape the nominal water demand: it can be done either by evaluating the 48% of the worst yearly pattern and diminishing the nominal water demand of these values, or by evaluating step by step the tributary inflow as the 48% of the Tono dam inflow and diminishing the water demand of this value. The second option was adopted in this study (see fig. 3.2) and $\beta(t)$ was set $\beta(t) = 1$ from May 3rd to June 1st, $\beta(t) = 0.8$ from June 2nd to September 4th, $\beta(t) = 0.3$ from September 5th to May 2nd; γ was set equal to 2.

Since the physical meaning of g_{t+1}^{irr1} is hardly interpretable, since it is expressed as $[(m^3/day)^2]$, other four more physically meaningful step-costs were introduced to support the policy evaluation and comparison. The first one is the daily deficit $[m^3/s]$:

$$g_{t+1}^{irr2} = (w_t - (r_{t+1} - q_{t+1}^{MEF}))^+ \tag{3.7}$$

The other three step-costs are defined in the same way, but over a shorter inter-

[3] The tributary contribution of 48% is calculated as the ratio between the catchment area of Tono dam (38.10 km^2) and the catchment area of the downstream tributary (19.72 km^2). This is a viable procedure as the two basins are very close to each other and are at similar heights, giving them similar meteorological conditions.

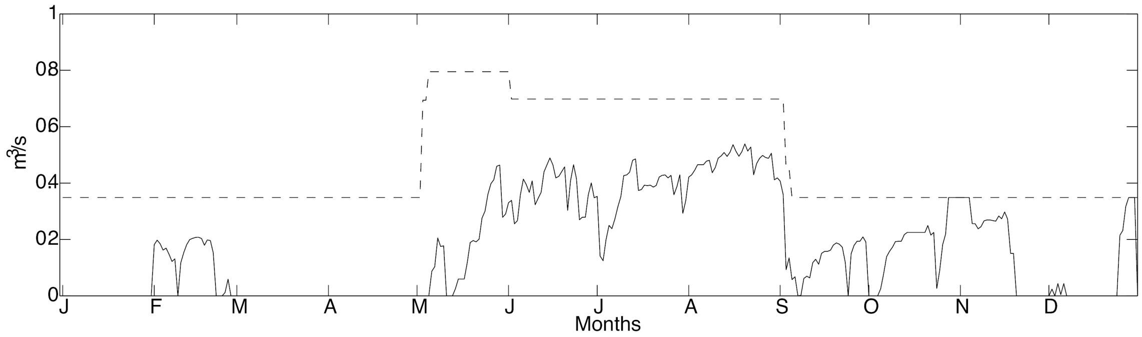


Figure 3.2: The nominal water demand (dashed line) and the one reshaped (continuos line) to take into account the contribution of the tributary to the downstream river.

annual period:

$$g_{t+1}^{irr3} = (w_t - (r_{t+1} - q_{t+1}^{MEF}))^+ \tag{3.8}$$

which is calculated over the winter period;

$$g_{t+1}^{irr4} = (w_t - (r_{t+1} - q_{t+1}^{MEF}))^+ \tag{3.9}$$

which is calculated in May;

$$g_{t+1}^{irr5} = (w_t - (r_{t+1} - q_{t+1}^{MEF}))^+ \tag{3.10}$$

which is calculated over the summer period.

Temperature

The step-cost for this sector is the squared daily difference between the temperature of the inflow and the temperature of the outflow, as follows:

$$g_{t+1}^{temp1} = (T_{t+1}^{out} - T_{t+1}^{in})^2 \tag{3.11}$$

where T_{t+1}^{in} is defined as

$$T_{t+1}^{in} = \frac{T_{t+1}^K a_{t+1}^K + T_{t+1}^F a_{t+1}^F}{a_{t+1}^K + a_{t+1}^F} \tag{3.12}$$

with T^K and T^F being the average temperature [°C] of the inflow between t and $t+1$ respectively for the Kango and Fukuro river, and a_{t+1}^K and a_{t+1}^F the corresponding inflow, while T_{t+1}^{out} is the average temperature in the same time interval in a section just downstream of the turbine outlet. By assuming negligible the effects of the turbines on the temperature as well as the temperature variation along the river course from the dam to that river section, T_{t+1}^{out} is given by

$$T_{t+1}^{out} = \frac{\sum_{i=1}^{N} T_{t+1}^{i} r_{t+1}^{i} + T_{t+1}^{spill} r_{t+1}^{spill}}{\sum_{i=1}^{N} r_{t+1}^{i} + r_{t+1}^{spill}} \tag{3.13}$$

where T_{t+1}^{i} is the average temperature between t and $t+1$ in the layer corresponding to the i-th controlled siphon and T_{t+1}^{spill} is the average temperature in the layer of the spillway. Also for this sector a more intuitive step-cost is defined as the daily difference of temperature between inflow to and outflow from Tono, as follows:

$$g_{t+1}^{temp2} = (T_{t+1}^{out} - T_{t+1}^{in}) \tag{3.14}$$

Figure (3.3) represents all the sectors affected by Tono dam operation, along with the corresponding evaluation criteria (the sectors that have not any associated criterion are represented with dotted line).

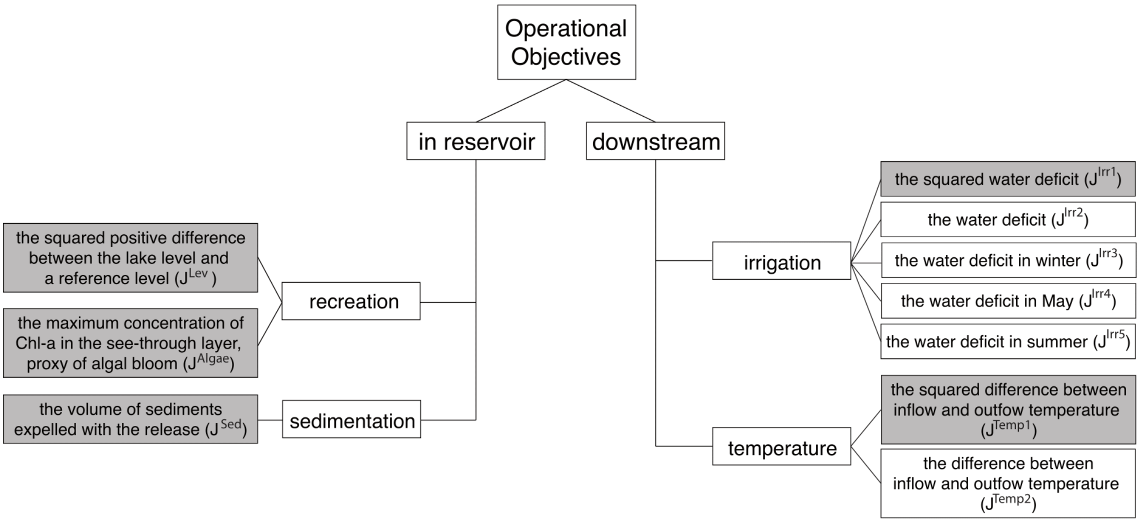

Operational Objectives
in reservoir
downstream
the squared positive difference between the lake level and a reference level (J^Lev)
the maximum concentration of Chl-a in the see-through layer, proxy of algal bloom (J^Algae)
recreation
the volume of sediments expelled with the release (J^Sed)
sedimentation
irrigation
the squared water deficit (J^Irr1)
the water deficit (J^Irr2)
the water deficit in winter (J^Irr3)
the water deficit in May (J^Irr4)
the water deficit in summer (J^Irr5)
temperature
the squared difference between inflow and outfow temperature (J^Temp1)
the difference between inflow and outfow temperature (J^Temp2)

Figure 3.3: The hierarchy of criteria for the Tono dam system

3.1.3 DYRESM-CAEDYM Model

In principle, a 3D spatially-distributed model of the hydrodynamic and ecological processes taking place in the lake should be considered. However, there are basically two reasons for not adopting this type of model:

1. The reservoir is being created by damming two rivers in a quite narrow section of their course and therefore longitudinal and vertical phenomena are dominating. A 2D water quality model (CE-QUAL-W2) should be enough for accurately describing the system.

2. The estimated simulation/real-time time ratio of a 3D model, such as ELCOM-CAEDYM, is 1/30 days, and this make it totally unsuitable for supporting the design of a release policy.

Since a 2D model of Tono dam was not available, the final choice was for a 1D model, namely DYRESM-CAEDYM. With this model, the spatial dynamics between the inlet and the outlet of the reservoir are lost, however the computational time drops off to nearly 1/12275 days. DYRESM (DYnamic REServoir Simulation Model, see Imerito (2007)) is an hydrodynamical model that is able to simulates the vertical distribution of temperature, salinity and density in lakes that can be monodimensionally approximated. The reservoir is represented by a set of horizontal layers that have different depth according to the amount of water accumulated into the reservoir. DYRESM is used together with CAEDYM (Computational Aquatic Ecosystem DYnamics Model, see Hipsey et al. (2006a)) which simulate chemical and biological processes. CAEDYM represents processes like C, N, P, Si and DO cycle, inorganic suspended solids, and phytoplankton dynamics.

The model is based on a Lagrangian architecture that models the reservoir as horizontal layers of uniform properties (i.e., temperature and water qualities). The thickness of the layers varies in time depending on the water density profile. In this study, the minimum and the maximum thickness of a layer is set to 1 m and 2 m, respectively, which correspond to allow the definition of more than 30 layers in the Tono Dam reservoir. Twenty-one state variables are defined for each layer, for a total of nearly 600 state variables (including the level). Because observational data are not yet available, the model was calibrated by applying the same parameter values as obtained in the calibration of a neighboring reservoir with similar feature and size (Castelletti et al., 2014).

3.2 Formulation of the optimal control problem

The problem of designing the optimal control policy p^* of the Tono dam can be formalized as an optimal control problem:

$$J^* = \min_p E_{\{\varepsilon_t\}_{t=1,\dots,h}}[J(\mathbf{x}_0^h, \mathbf{u}_0^{h-1}, \boldsymbol{\varepsilon}_1^h)] \tag{3.15a}$$

subjected to

$$\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\varepsilon}_{t+1}) \ \textit{t=0,1,...,h-1} \tag{3.15b}$$

$$m_t(\mathbf{x}_t) = \mathbf{u}_t \in \mathcal{U}_t(\mathbf{x}_t) \ \textit{t=0,1,...,h-1} \tag{3.15c}$$

$$\boldsymbol{\varepsilon}_{t+1} \sim \phi_t(\cdot) \ \textit{t=0,1,...,h-1} \tag{3.15d}$$

$$\mathbf{x}_0 \ \textit{given} \tag{3.15e}$$

$$p \triangleq \{m_t(\cdot); t = 0, 1, ..., h-1\} \tag{3.15f}$$

$$\textit{any other constraints t=0,1,...,h-1} \tag{3.15g}$$

where $J = \sum_{j=1}^{n} \lambda^j J^j$ ($\sum_{j=1}^{n} \lambda^j = 1$ with $\lambda^j \geq 0 \ \forall j$) is a weighted sum of n design criteria and $\mathbf{x}_t$ is the reduced state vector[4].The vector of control variables $\mathbf{u}_t$ contains N release decision corresponding to N different outlets. The natural choice would be to consider all the 15 siphons, including the last available one at 18 m from the bottom (bottom outlet) and the intake in the sediment storage at 11 m from the bottom (sediment outlet). However, the computational time required to design such a complex control policy overcomes the objectives of the project and a simplification was unavoidable. The more effective release decision variables are the release decision u_t^{-3}, u_t^{-7}, u_t^{-13}, and u_t^{bot} that provide the water volumes to be released between t and $t+1$ respectively from the outlets at -3 m, -7 m, -13 m

[4] State reduction techniques are applied in order to make the problem computationally tractable: the underlying idea is that not all the dynamic modes described by the 1D DYRESM-CAEDYM model are equally relevant in the cause-effect relationship linking release decision to evaluation criteria. The original state vector can be therefore reduced to a smaller vector which is still significant in conditioning the release decision. Both formal and empirical (expert-based) approaches can be adopted to perform such reduction.

depth with respect to the water surface, and the volume released from the bottom outlet. The underlying idea is that these water elevations should correspond, in the average reservoir conditions, to the epiliminium, the thermocline region and the hypolimnium of the stratified reservoir. The decision variables are defined over a feasibility set $\mathcal{U}_t(\mathbf{x}_t)$ that takes into account which outlets are available given the storage, the physical constraints imposed by the siphons and the SWS outlet size, and the hydraulics of the SWS. More precisely, each siphon cannot convey more than 7.353 m^3/s, while the flow rate allowed by the SWS outlet is 13.780 m^3/s. The water volume released through each siphons cannot be freely decided, but depends on the total amount released from the SWS, which is hydraulically equally divided among the open siphons. Notice that when more than one siphon is opened, each siphon cannot be operated at the maximum capacity. Finally, $\boldsymbol{\varepsilon}_{t+1}$ is the vector of stochastic disturbance (e.g. inflow, wind, solar radiation, nutrient load in the inflow etc.).

The problem resolution is computationally intractable due to the dimensionality of the state-action space, i.e., having hundreds of state variables, 4 controls, and the high number of objectives considered. A linear increase in the number of objectives indeed yields a factorial growth in the number of problems to solve, namely a four objective problem requires to solve also 4 single-objective problems, 6 two-objective problems, and 4 three-objective problems (Reed and Kollat, 2013; and Giuliani et al., 2014). Therefore, state reduction is necessary to make the problem computationally tractable.

In previous works (Castelletti et al., 2014; Giuliani et al., 2014), an expert-based state reduction was performed to select a 6-variable state vector, which comprises the following variables: time, reservoir storage, T-3, T-13, TSS-3, TSS-13. In this study, the state reduction is performed using a formal approach based on Dynamic Emulation Modeling (DEMo) (Castelletti et al., 2012a).

Chapter 4

Results

The purpose of this study is to compare two classes of Dynamic Emulation Modeling (DEMo): selection-based, and projection-based techniques. First, the preliminary steps to model-reduction are described, namely (i) the design of experiments, (ii) expert-based variable aggregation, and (iii) a reduction of sample-size by random sampling. Then, the competing techniques, Recursive Variable Selection, and Principal Component Analysis are applied on the resulting data-set $\tilde{\mathcal{F}}$ to reduce the number of variables. Finally, emulators are created for three outputs: two in-reservoir water quality outputs; algal bloom, represented by the daily average hourly maximum concentration of Chlorophyll-a in the see-through layer, and the daily total suspended solid; and one downstream water quality output, i.e. the squared daily difference between the temperatures of the inflow and the outflow. The outputs will denoted as g_{t+1}^{temp}, g_{t+1}^{sed}, and g_{t+1}^{algae} respectively, and their associated step-costs are found in equations 3.11, 3.4, and 3.3. These three represent water quality issues, which require the use of complex, process-based models to describe the hydrodynamic and ecological processes involved. Instead, the discarded objectives are associated to water quantity, and their associated emulators are somewhat trivial as they mainly depend on storage, time, and possibly inflow volumes. The chapter is divided as follow: Part I includes the pre-processing steps to applying RVS, PCA, SPCA, or WPCA, whereas Part II discusses the results from the four approaches separately. Finally, the four approaches are compared with respect to different criteria in Part III.

4.1 Part I: Perquisites to model reduction

4.1.1 Design of experiments

Like any data-driven emulation exercise, the experiment starts with running a sequence of computer simulations of the original model $\mathcal{M}$ aimed at creating a dataset $\mathcal{F}$ to be utilized for the emulation. In the case of the Tono dam case study, this was done by running simulations on the 1D model, DYRESM-CAEDYM. The model was simulated under 100 pseudo-random sequences of controls, sampled from an irregular grid with lower probability assigned to high release values in order to reduce the occurrence of full reservoir drawdown. The model consists of an exogenous driver vector $\mathbf{W}_t$ which includes 50 components, accounting for the main hydro-meteorological processes and water pollution loads. The control vector $\mathbf{u}_t$ is assumed to have four components (i.e., the release decisions from the siphons $\mathbf{u}_t^{-3}$, $\mathbf{u}_t^{-7}$, $\mathbf{u}_t^{-13}$, and $\mathbf{u}_t^{bot}$ $[m^3/s]$). The trajectories of $\mathbf{u}_t$ are designed to span as much as possible the state-control space. As for $\mathbf{W}_t$, the time series of observational data over the period 1995-2006 are available, and, considering the variety of conditions included in them and the length of the series, they are directly used without further data generation.

For each of the 100 simulation scenarios so obtained, the coupled DYRESM-CAEDYM model is run with 1 m vertical grid resolution and a simulation step of 1 min. The simulated data, sampled with a time-step Δt equal to one day, are finally stored in the data-set $\mathcal{F}$ of tuples $\{\mathbf{X}_t, \mathbf{W}_t, \mathbf{u}_t, \mathbf{X}_{t+1}, \mathbf{Y}_t\}$, with dimensionality N_x, N_w, N_u and N_y equal to $\sim 10^3$, 50, 4 and 3. The dimensionality N_y of $\mathbf{Y}_t$ is equal to 3 since only three immediate costs (outputs) will be considered for emulation: g_{t+1}^{temp}, g_{t+1}^{sed}, and g_{t+1}^{algae}. The 100 simulation runs, each one with a simulation horizon of 12 years, give a total of $\sim 4.50 \times 10^5$ tuples.

4.1.2 Variable aggregation

The dimensionality of the dataset $\mathcal{F}$ is unsuitable for the model-reduction techniques used in this study, specially for selection-based techniques, as they tend to be computationally demanding. In order to create a lower dimension dataset, *variable aggregation* is applied, whose aim is to transform $\mathbf{X}_t$ and $\mathbf{W}_t$ into two lower dimension vectors $\tilde{\mathbf{X}}_t$ and $\tilde{\mathbf{W}}_t$, with dimensionality $\tilde{N}_x \ll N_x$ and $\tilde{N}_w \ll N_w$ by using a suitable aggregation scheme. Castelletti et al. (2012b) employed an expert-based aggregation scheme for $\mathbf{X}_t$, thereby reducing the original $\sim 10^3$ components to 19. The aggregation was made such that the states most relevant to characterize the dynamic behavior of the reservoir with respect to the management objectives were kept. For example:

- dissolved oxygen DO $[mg\ O/L]$, ammonium NH_4 $[mg\ N/L]$, nitrate NO_3 $[mgN/L]$, phosphate PO_4 $[mgP/L]$, silicium SiO_2 $[mgSi/L]$, pH and chlorophyll-a $chlor$ $[mg\,Chla/L]$ in both rivers; Kango and Fukuro.

- the reservoir temperature T^i °C], and total suspended solid tss^i $[g/m3]$, in the layers located i meters below the surface, with i = 3, 7, 13, b, s (b and s are the layers corresponding to the bottom and sediment outlet).

- the reservoir level h $[m]$ and storage s $[m^3]$;

- the level $h^{T_{in}}$ $[m]$ at which the water temperature T in the reservoir equals the average temperature T_{in} °C] of the inflows and the level $h^{tss_{in}}$ $[m]$ at which the total suspended solid tss in the water column equals the average total suspended solid tss^{in} $[g/m3]$ of the inflows;

- the maximum values $T^{max} = \underset{h}{max}\, T(h)$°C] and $tss^{max} = \underset{h}{max}\, tss(h)[g/m3]$ of the temperature and total suspended solid over the water column;

- the level $h^{T_{max}}$ and $h^{tss_{max}}$ corresponding to T^{max} and tss^{max}, i.e. $arg\underset{h}{max}T(h)$ and $arg\,\underset{h}{max}\,tss(h)$;

This expert-based aggregation scheme reflects the idea of extracting from the state vector $\mathbf{X}_t$ the features that might be most relevant to characterize the dynamic behavior of the controlled reservoir with respect to the different management objectives. However, not all the states in $\tilde{\mathbf{X}}_t$ are in a causal relationship with the considered objectives (g^{temp}_{t+1}, g^{sed}_{t+1}, and g^{algae}_{t+1}). As for the exogenous vector $\mathbf{W}_t$, variable aggregation was not considered, as these variables are already lumped in space. For a full list of candidate input variables, see table 4.1.

4.1.3 Sample-size reduction

In addition to the dimensionality reduction described in the previous section, a reduction of the number of observations was also considered. This was done for two reasons:

Table 4.1: List of candidate input variables

#	Name	#	Name	#	Name	#	Name	#	Name
Kango inflow data		**Fukuro inflow data**		**Meteorological Data**		**State Variables**		**Control variables**	
1	Volume	23	Volume	45	SW	51	t	70	u-spill
2	Temperature	24	Temperature	46	Cloud cover	52	h	71	u-3
3	Salinity	25	Salinity	47	Air Temperature	53	s	72	u-7
4	NH4	26	NH4	48	Vap. Press	54	Taff	73	u-13
5	NO3	27	NO3	49	Wind Speed	55	hTaff	74	u-bot
6	PONL	28	PONL	50	Rain	56	gateTaff	75	u-sed
7	PO4	29	PO4			57	TSSmax		
8	POPL	30	POPL			58	hTSSmax		
9	DO	31	DO			59	gateTSSmax		
10	DOCL	32	DOCL			60	T-3		
11	POCL	33	POCL			61	T-7		
12	SSOL1	34	SSOL1			62	T-13		
13	SSOL2	35	SSOL2			63	Tbot		
14	SSOL3	36	SSOL3			64	Tsed		
15	SSOL4	37	SSOL4			65	TSS-3		
16	SSOL5	38	SSOL5			66	TSS-7		
17	SSOL6	39	SSOL6			67	TSS-13		
18	pH	40	pH			68	TSSbot		
19	SiO2	41	SiO2			69	TSSsed		
20	CYANO	42	CYANO						
21	CHLOR	43	CHLOR						
22	FIDAT	44	FIDAT						

Note: For full variable notation, see the CAEDYM science manual (Hipsey et al., 2006b).

1. The Iterative Input Selection algorithm within RVS is a time-intensive procedure, and the time required to complete a single iteration increases super-linearly with the number of observations (Appendix A). Therefore, a balance must be found between the computing requirements of RVS and number of observations in the dataset.

2. The purpose of creating these emulation models is to ultimately use them to speed-up the procedure of designing the optimal operating policy of the reservoir. This makes it necessary to take into account the time-effect of sample-size from the beginning, so as not to impair the time-saving benefit of variable-reduction.

As for principal component analysis, it turns out that computational time is not critical with respect to the number of observations in this problem. However, a unified sample-size was adopted in both techniques to provide a fair comparison of the results. According to preliminary experiments (Giuliani, 2010), it was decided that 10% of the original observations, i.e., $\sim 4.50\times10^4$ is a sufficient sample-size. The sufficiency was proven empirically for both RVS and PCA. The sample is obtained by taking a random subset of $\mathcal{F}$ subject to the constraint of maintaining the same frequency of use of each outlet as in the original DOE.

4.2 Part II: Model-order reduction

4.2.1 Recursive Variable Selection

In this section, the RVS results are reported. The results were obtained following the RVS algorithm described in chapter 4 and appendix A.3, which in turn uses Iterative Input Selection (see A.4) as an input selection algorithm. The state-variables for which IIS was performed are items 54 to 69 in table 4.1, which are water quantity and quality state-variables. For states t(time), h_t(water level), and s_t(storage), IIS was not needed, because the causal networks for these states are known a priori. Time depends on the previous time-step, while storage is governed by the following equations:

$$s_{t+1} = s_t + a_{t+1} - E_{t+1} - r_{t+1}, \tag{4.1a}$$

$$E_{t+1} = e_{t+1} S(s_t), \tag{4.1b}$$

where a is the inflow, E is the evaporated volume, S is the volume of the water surface, e is the specific evaporation volume, and r is the released volume, which coincides with the control variables u_t in this problem. Evaporation was considered negligible. Finally, h_{t+1} is proportional to s_{t+1}.

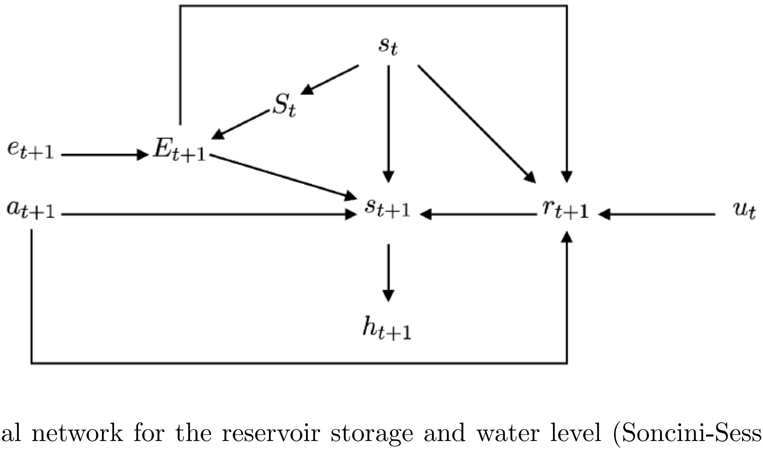


Figure 4.1: Causal network for the reservoir storage and water level (Soncini-Sessa et al., 2007a: Ch. 4)

As mentioned in section 2.2.3, IIS utilizes *Extra-tress* (Ernst et al., 2005) as a model building (MB) algorithm. Extra-trees are characterized by three parameters: M, K, and n_{min}. M is the number of trees in the ensemble, K is the number of alternative cut-directions (i.e. the number of candidate variables), and n_{min} is the minimum-cardinality for splitting a node. The parameter values were chosen based on guidelines in (Geurts et al., 2006), and empirical experiments in (Castelletti et al., 2011, 2012b), and adjusted to fit the time constraint of this experiment. "The higher the value of M, the better from the accuracy point of view" (Geurts et al., 2006). M was set to 500 for IIS on outputs to achieve the highest accuracy possible, and it was reduced to 100 for state-variables to balance accuracy with computational time, as there are more states than outputs. K is set to 75, which is the number of candidate variables, and n_{min} is set to 2, following the guidelines from the aforementioned studies. The stopping criterion for IIS in this experiment is based on the R^2 performance between the measured output and the output of the underlying model (in k-fold cross-validation[1], with $k = 10$). The results of the IIS ranking (reported in appendix B) show all the selected variables for each output or state-variable with their associated relative δR^2 (i.e. the difference in R^2 performance contributed by a selected variable, normalized by the total R^2 of all selected variables). In the next section, different causal networks are constructed by varying a threshold on this criterion.

Causal Networks

The RVS algorithm proceeds by first applying IIS on the output (g_{t+1}^{temp}, g_{t+1}^{sed}, or g_{t+1}^{algae}). This run of IIS identifies the set of input variables most relevant in explaining the process-based model output. From the first set of selected variables, those that are state-variables need to have their dynamic behaviors explained. Therefore, subsequent runs of IIS are performed on the selected states and the newly selected variables and states are added to the causal network of the output. This procedure continues until no states that have not been selected before are added, and the output causal network becomes complete.

Different causal networks for a single output can be constructed by setting a threshold on either the R^2 contribution of the variable or the number of selected

[1] This means that the data-set is divided in k parts (folds), and for each one both calibration and validation are performed. The final estimate of the model performance in calibration/validation is obtained by averaging the values associated to each fold.

variables from each IIS occurrence. For instance, RVS could be performed such that only variables with a relative R^2 contribution of more than 2% in IIS are passed onto the next iteration of RVS. Alternatively, a threshold could be set on the number of variables taken from each IIS ranking, e.g., the first 3 or 5 variables by ranking order. The former method is implemented here; the next sections show how the complexity of the resulting causal networks varies with the threshold on R^2.

Temperature (g_{t+1}^{temp})

Table 4.2: Causal network complexity for g_{t+1}^{temp}

min δR^2	number of inputs (states, controls, exogenous)
0	44 (19, 4, 21)
0.5	23 (8, 4, 11)
1	20 (7, 4, 9)
1.5	18 (6, 4, 8)
2	16 (6, 4, 6)
2.5	15 (16, 4, 5)
3	14 (5, 4, 5)
3.5	14 (5, 4, 5)
4	9 (2, 4, 3)
4.5	9 (2, 4, 3)
5	9 (2, 4, 3)

Table 4.2 shows how the size of the network changes with different values of the minimum δR^2 threshold. When the threshold on min δR^2 is set to 0, i.e., all selected variables are included, the network consists of 44 variables, including 19 states, while the total number of candidate inputs is 75. Yet, this network is too complex, and does not represent the desired degree of model-reduction. The network complexity decreases with the increase of min δR^2. The change occurs in the number of states and exogenous drivers, whereas even with the strictest threshold min ($\delta R^2 > 5\%$), all four control variables (u_t^{-3}, u_t^{-7}, u_t^{-13}, and u_t^{bot}) remain in the network. They are connected to g_{t+1}^{temp} directly, and through state-variable h_t, the water level, which explains 9.1% of the output variance, according to table B.1. This indicates that water temperature is controllable using the four siphon outlets, and that they play an important part in determining the dynamic behavior of this objective. Some examples of the causal networks are visualized in the figures 4.2, 4.3, and 4.4, which have a threshold on min δR^2 of 2%, 3%, and 5%, respectively.

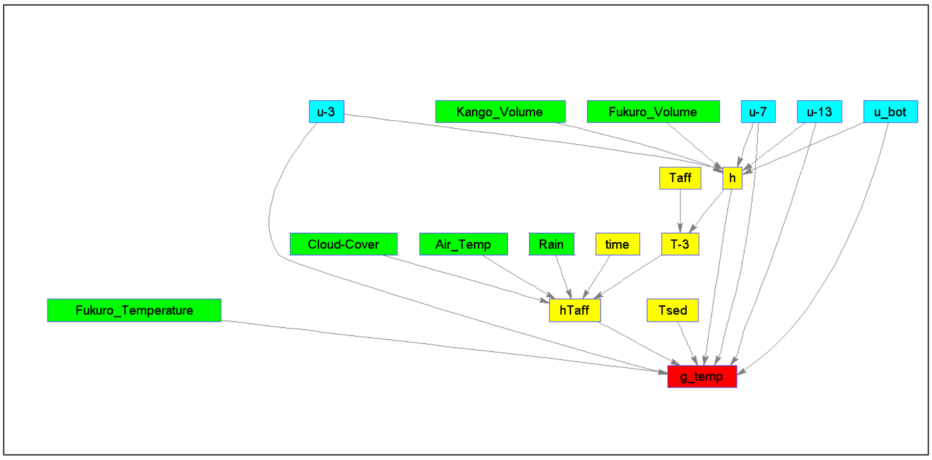


Figure 4.2: Causal network for g_{t+1}^{temp}, min relative $\delta R^2 > 2\%$

At $\delta R^2 > 2\%$, the output connects directly to: 2 state-variables, one exogenous input, and all four control variables.

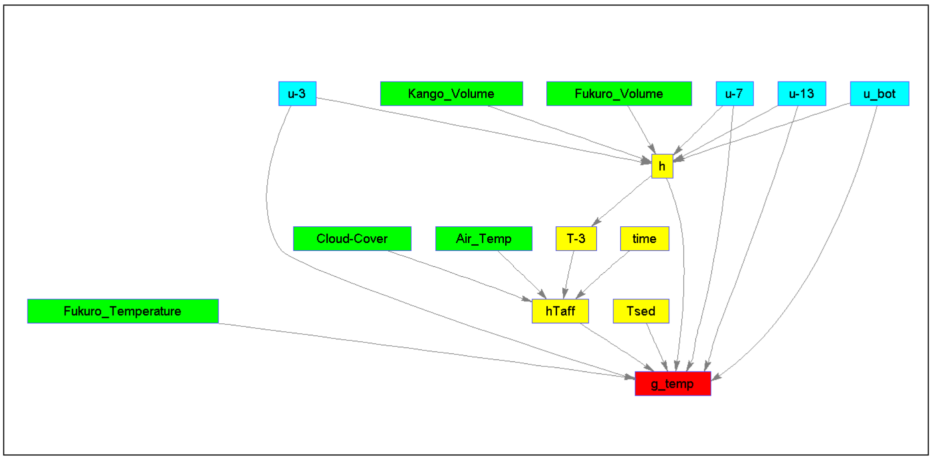


Figure 4.3: Causal network for g_{t+1}^{temp}, min relative $\delta R^2 > 3\%$

At $\delta R^2 > 3\%$, the outcomes of the first two iterations of RVS remain the same, i.e. at the output level and the states connected directly to the output, while in the third iteration, the number of states and exogenous inputs each drop by one. Namely, T_t^{aff} and $Rain_t$ are no longer selected.

Finally, at $\delta R^2 > 5\%$ (see fig. 4.4), the size of the network is reduced significantly, and it contains only two state-variables; h_t (water level) and T_t^{sed} (water temperature at sediment level), indicating their importance in explaining g_{t+1}^{temp} relative to the other state-variables. They are connected to the reservoir stratification conditions, which depend on the water level (the higher the level is, the higher the stratification)

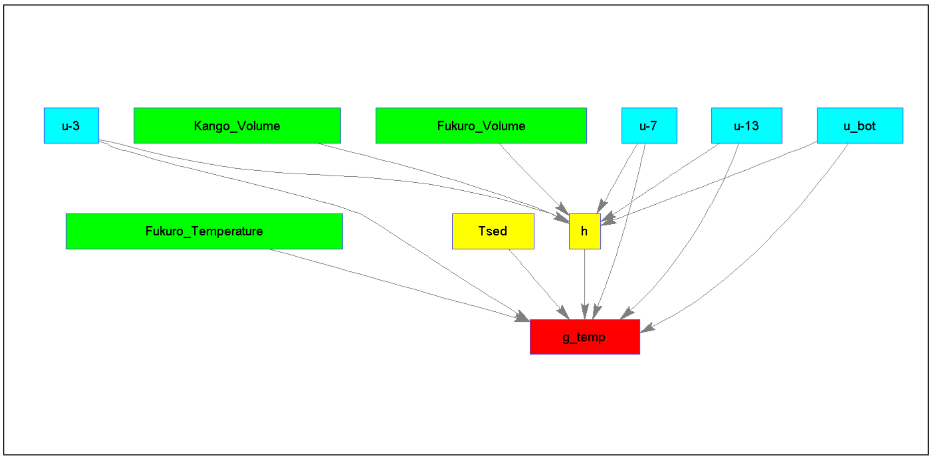


Figure 4.4: Causal network for g_{t+1}^{temp}, relative $\delta R^2 > 5\%$

(Castelletti et al., 2012b). To a lesser extent, time is an important driving force, and it represents the annual periodicity of the process being modeled, while T_t^{-3} (temperature at -3 meter) serves the same purpose as h_t and T_t^{sed}. The most important exogenous driver; the temperature of Fukuro, can be seen as a proxy of the average inflow temperature (of Kango and Fukoro).

Sediments (g_{t+1}^{sed})

Table 4.3: Causal network complexity for g_{t+1}^{sed}

min δR^2	number of inputs (states, controls, exogenous)
0	4 (1, 1, 2)
0.5	4 (1, 1, 2)
1	3 (1, 0, 2)
1.5	3 (1, 0, 2)
2	3 (1, 0, 2)
2.5	3 (1, 0, 2)
3	3 (1, 0, 2)
3.5	3 (1, 0, 2)
4	3 (1, 0, 2)
4.5	3 (1, 0, 2)
5	3 (1, 0, 2)

Table 4.3 shows that there are only two distinct causal networks for g_{t+1}^{sed} as min δR^2 is varied from 0% to 5%. The size of the networks is significantly smaller

when compared to g_{t+1}^{temp}. According to table B.2, The exogenous driver Kango POCL (Labile Particulate Organic Carbon) explains 94% of the output R^2, while the second selected variable, the state TSS-3 (Total suspended solids at -3 meters), explains around 5%. Auxiliary runs of IIS performed on different resamples of the dataset revealed that the first selected variable varies at each run between a number of exogenous drivers including (Fukoro SSOLi, Kango SSOLi, where $i = 1, \ldots, 6$; and Kango or Fukoro POCL). SSOLi (suspended solids) and POCL variables represent different classes of suspended particles. Any of them may act as a proxy of all suspended solids flowing into the reservoir, hence its importance in explaining the dynamics of the released sediments. Moreover, this objective can be controlled by the siphon at -3 meters. Figures 4.5 and 4.5 show a visualization of the two causal networks for g_{t+1}^{sed}, with the control action appearing in the first one only, when all selected variables are included (min $\delta R^2 > 0\%$).

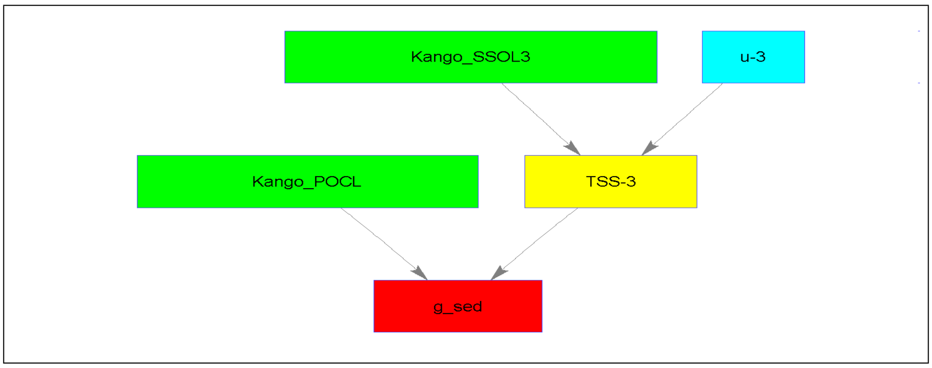


Figure 4.5: Causal network for g_{t+1}^{sed}, relative $\delta R^2 > 0\%$

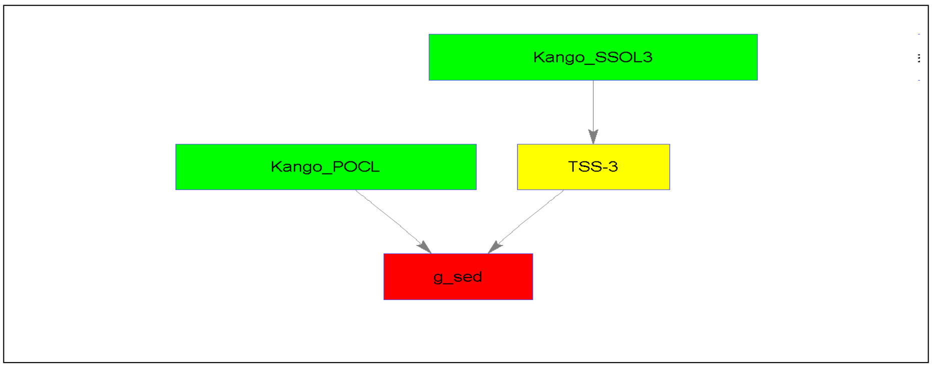


Figure 4.6: Causal network for g_{t+1}^{sed}, relative $\delta R^2 > 1\%$

Algae (g_{t+1}^{algae})

Table 4.4: Causal network complexity for g_{t+1}^{algae}

min δR^2	number of inputs (states, controls, exogenous)
0	43 (19, 4, 22)
0.5	25 (10, 4, 11)
1	11 (3, 4, 4)
1.5	11 (3, 4, 4)
2	11 (3, 4, 4)
2.5	11 (3, 4, 4)
3	9 (2, 4, 3)
3.5	9 (2, 4, 3)
4	9 (2, 4, 3)
4.5	9 (2, 4, 3)
5	9 (2, 4, 3)

Table 4.4 shows four causal networks for g_{t+1}^{algae} (Chlorophyll-a concentration $[g_{t+1}/m^3]$) as min δR^2 is varied. The first two networks have 43 and 25 input variables, and 19 and 10 states, respectively. This makes them unsuitable for this model reduction exercise. The other two networks (min $\delta R^2 > 1\%$ and $> 3\%$) have 11 and 9 inputs; 3 and 2 states, respectively. The most significant state in explaining the output is time, which accounts 55% of its variance (See table B.3). This result has a physical meaning, since Chlorophyl-a concentration is a proxy of algal bloom, which tends to follow an annual pattern. The second variable; NH4 (ammonium) concentration in the Kango river, contributes $\delta R^2 = 27.8\%$ to the model. This variable represents a concentration of nutrients in the inflow that algae consume. The third variable in importance is the water level which contributes $\delta R^2 = 10\%$. Level plays an important factor since g_{t+1}^{algae} is defined to measure the concentration of Chlorophyll-a just in the sea-through layer (7 m below surface; See 3.1.2), and high water levels represent favorable conditions for the reservoir stratification, trapping the nutrients needed by the algae in the shallow layers. Moreover, the dependence on level makes the output controllable. Figures 4.7 and 4.8 show a visualization of the two causal networks.

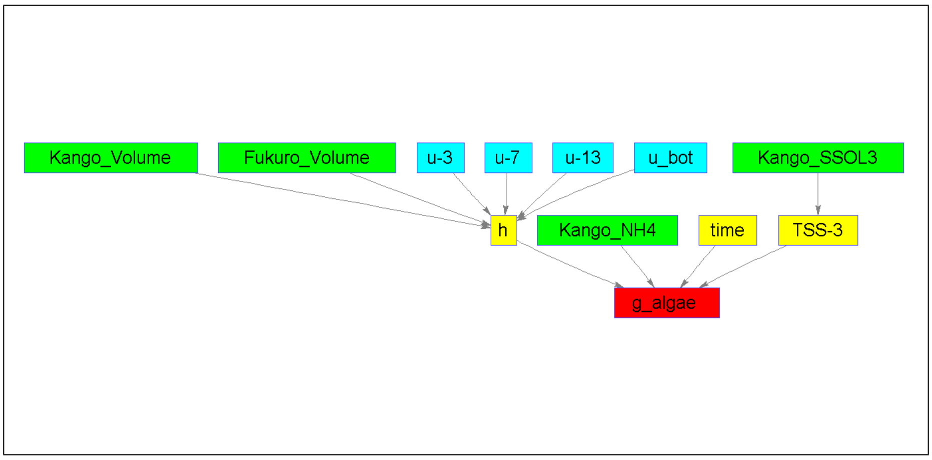


Figure 4.7: Causal network for g_{t+1}^{algae}, relative $\delta R^2 > 1\%$

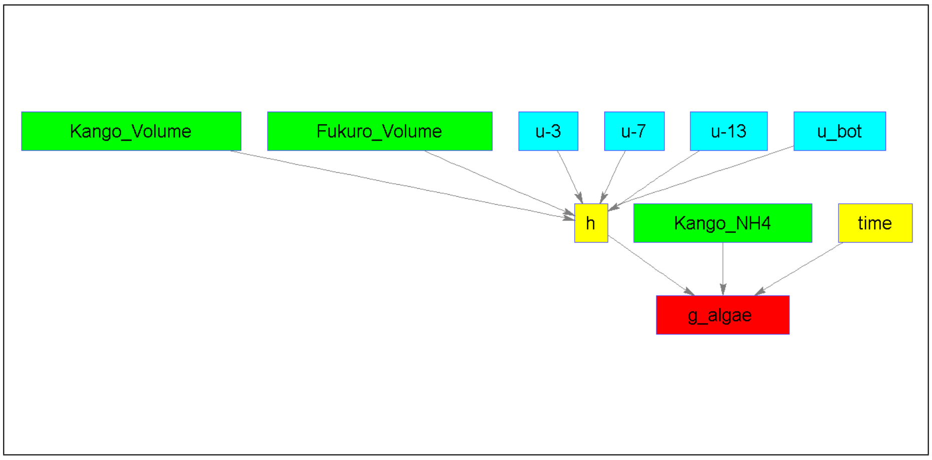


Figure 4.8: Causal network for g_{t+1}^{algae}, relative $\delta R^2 > 3\%$

In summary, it was shown that RVS was able to significantly reduce the dimensionality from the original 75. More specifically, the state-vector dimension is reduced from 19 down to 2 or 3 states for g_{t+1}^{temp} and g_{t+1}^{algae}, and a single state for g_{t+1}^{sed}. Physical interpretability is not compromised in this class of model-reduction techniques. In Part III of this chapter, the accuracy of the models built with variables selected in this section is assessed and compared to those made with principal component analysis.

4.2.2 Principal Component Analysis

In this section, the model-reduction problem is solved by implementing principal component analysis on dataset $\tilde{\mathcal{F}}$. All input variables of the data set are projected into principal components, from which a sufficient subset is selected to build an emulator.

Adequate sample-size

The first step of the analysis aims to assess if the reduced sample-size ($\sim 4.5 \times 10^5$ observations) is adequate for PCA. This can done by analyzing the *stability* of loading vectors resulting from applying PCA to different sample sizes. A loading vector (and its corresponding principal component) are considered stable for a particular sample size if the absolute values and signs of the loadings do not vary significantly between different resamples.

Kocovsky et al. (2009) assessed this stability by analytically measuring the variation of the sign and magnitude of the loadings in the first 3 loading vectors, and the correlation between a loading vector, and its copies derived from different resamples. In this study, since too many loading vectors need to be analyzed, the assessment of stability is performed visually using heat maps. A heat map of loading vectors of PCA using all available observations ($\sim 4.5 \times 10^5$) is shown in figure 4.9; the figure displays how much the original variables load in absolute value on all PCs; each column represents a single loading vector, and each row represents one of the original variables. The color of each cell represent the absolute value by which an original variable loads on a loading vector.

Since this heat map shows the overall structure of the loadings when all observations are used, it can be used as a reference to judge the structure of loadings from smaller sample sizes, namely, if a smaller sample-size produces a similar loading structure, it is considered equivalently sufficient for PCA.

Loading heat maps were obtained from different sample-sizes ranging from 20 to 2×10^5, which cover a wide range of observation to variable ratios from less than 1 to ($\sim 2 \times 10^4$). Ten resamples of each size were taken. It was found that the structure of the loadings does not vary significantly for sample sizes larger than 1×10^5. Moreover, the structure remains stable among different resamples of the same size. In fact, the structure loses stability marginally for smaller samples, but fully deteriorates for samples with observation to variable ratios of less than 1. This result is consistent with the commonly given guidelines for PCA sample-sizes.

Figure 4.16 shows the loading heatmaps for the first 20 PCs from four resamples of the sample-size used in this experiment, i.e., 4.5×10^5 or 10% of the full sample given by the design of experiment stage. It can be noted that the loading structure for the first 10 PCs is almost identical across different resamples. For PCs 10 to 13, slight variability exists in the loadings of variables 71 to 74 (the control variables). Therefore, this sample size is judged to be sufficient for PCA, and a single sample of this size is chosen for the analysis, namely, the same sample used for RVS.

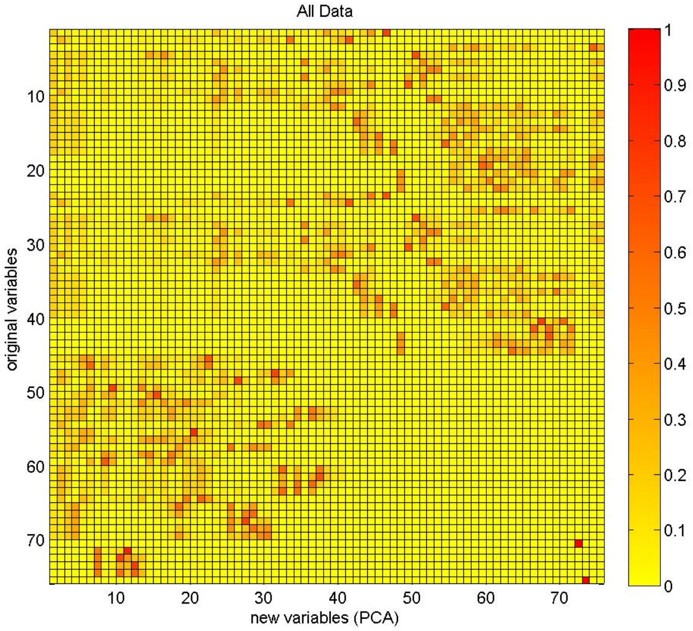


Figure 4.9: Loading vector heat map using the original sample-size

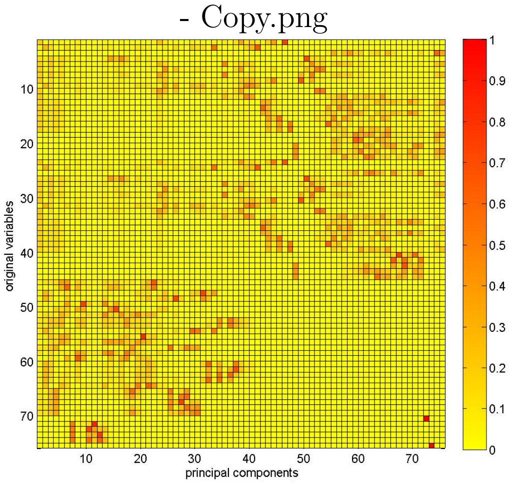


Figure 4.10: Heat map, full sample

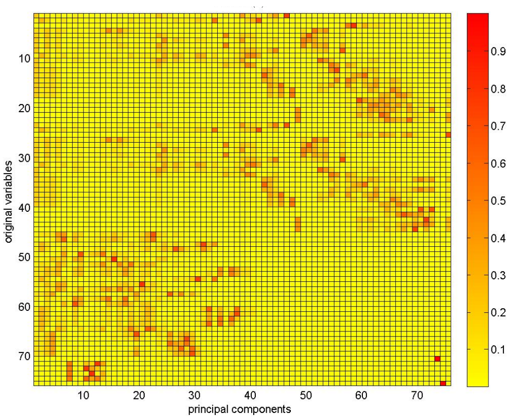


Figure 4.11: Heat map, sample size 300000

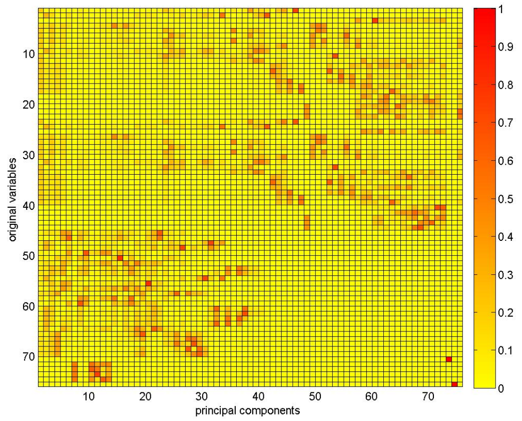


Figure 4.12: Heat map, sample size 200000

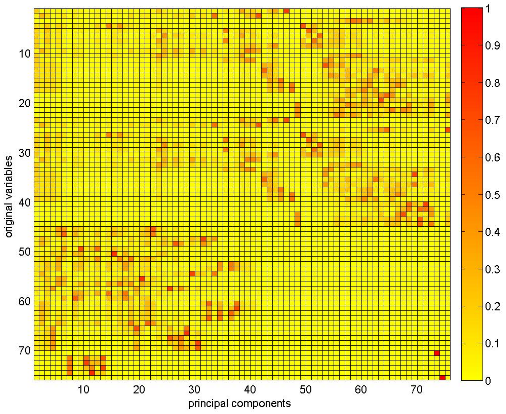


Figure 4.13: Heat map, sample size 50000

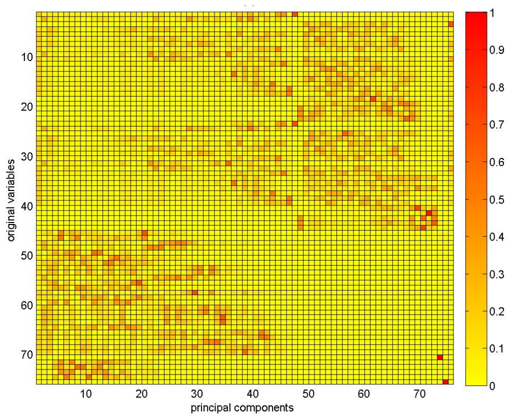


Figure 4.14: Heat map, sample size 75

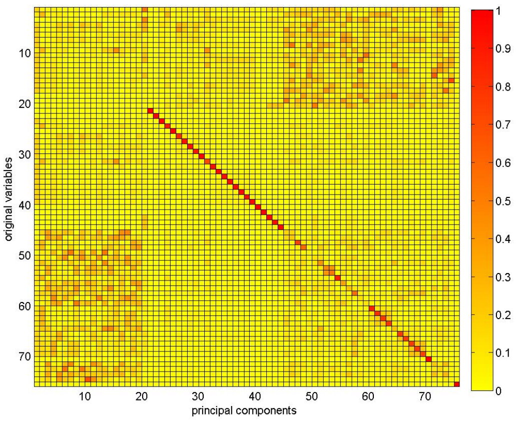
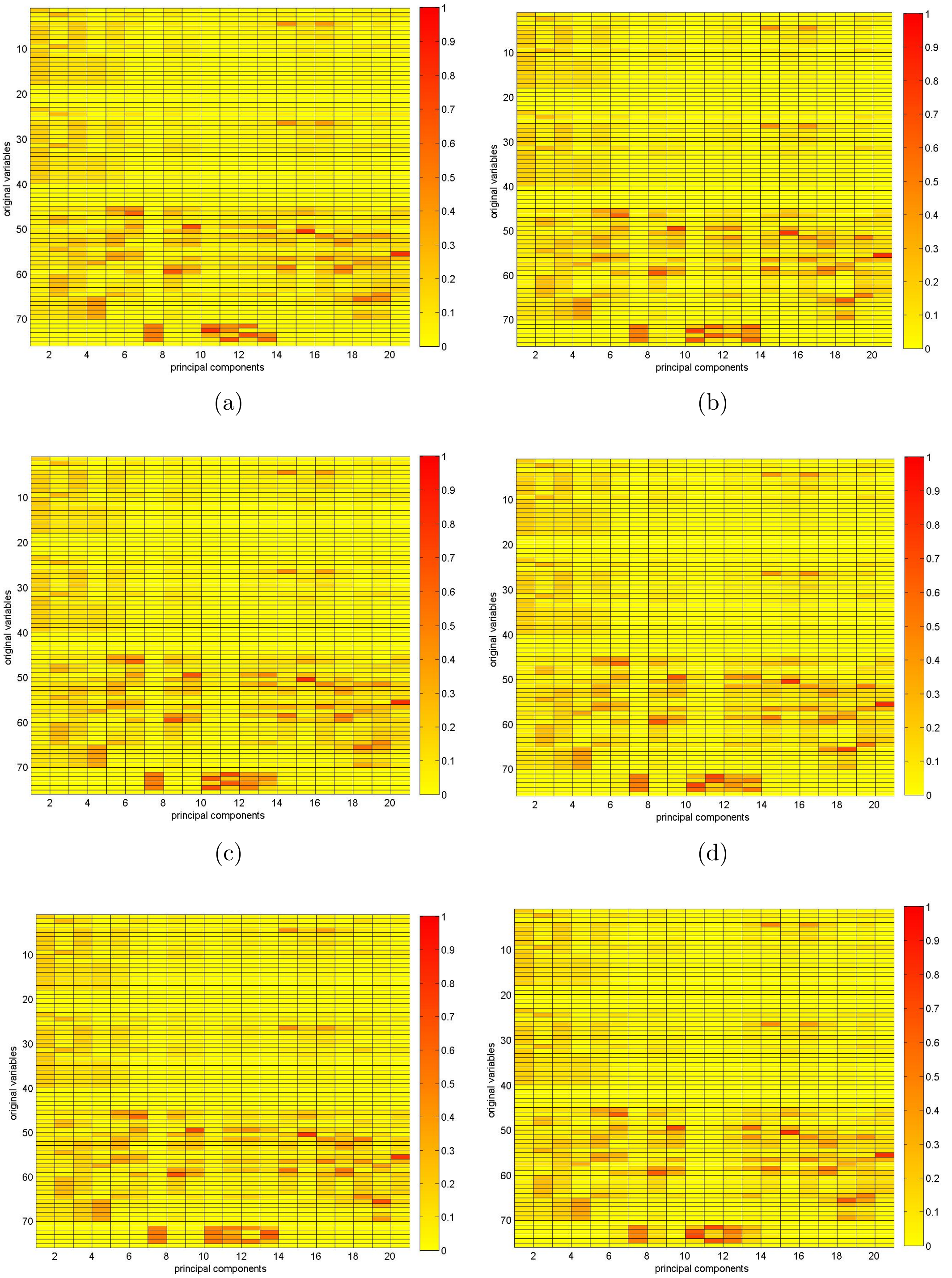


Figure 4.15: Heat map, sample size 20



(a) (b)

(c) (d)

(e) (f)

Figure 4.16: First 20 PCs loading heat maps from multiple resamples of size 45000

Explained Variance

Principal component analysis is performed on dataset $\tilde{\mathcal{F}}$ according to the procedure in section 2.3.The total variance of the dataset is explained by each principal component is given by λ_j/p, where λ_j is the eigenvalue corresponding to the the j-th PC, and p is the total number of variables, equal to 75 in this case.

It is found that the first principal component explains around 42% of the total variance, while the second to fifth components explain 19%, 7.8%, 5.3%, and 3.47% respectively. The first four PCs collectively explain around 75% of the total variance, while 90% and 99% of the variance are explained by the first 12 and 23 variables respectively. The distribution of variance over the fist 25 PCs is shown in figure 4.17.

The results show that PCA achieves good compression of the variance into fewer components, as expected. However, it is not straightforward to determine how many PCs to retain based only on explained variance, since different PCs might be relavant/irrelevant depending on the output variable of interest. The choice of PCs will be addressed in Part III of this chapter.

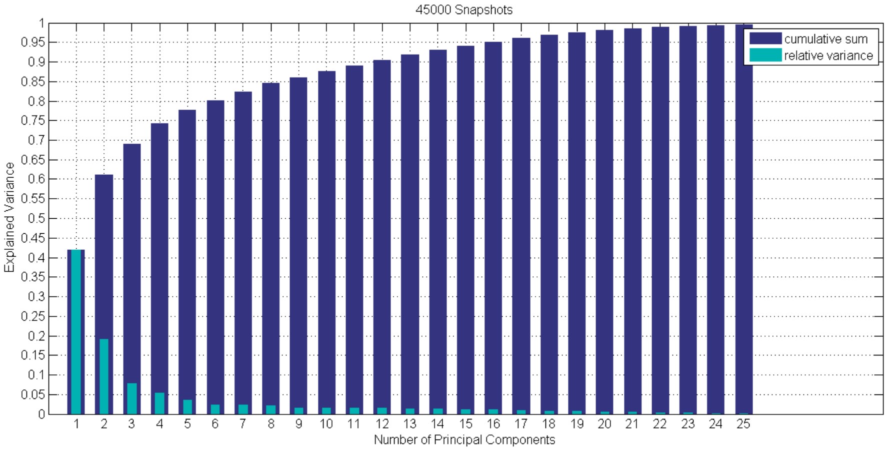
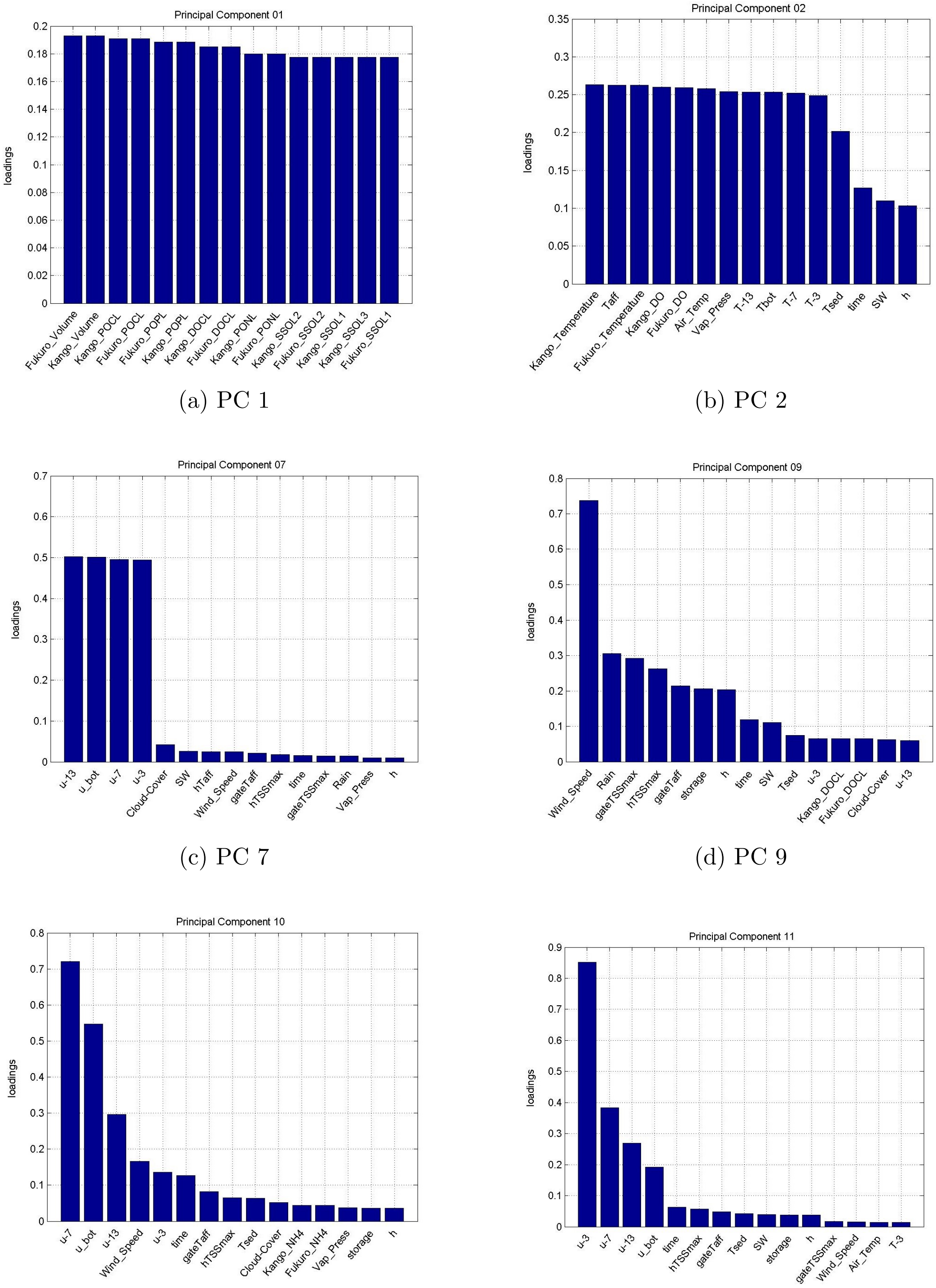


Figure 4.17: Relative and cumulative sum of the explained variance of the first 25 components.

Physical Interpretation

For ordinary PCA, it is useful for the interpretability of some PCs to either examine the loadings heat map(Fig. 4.9) or to rank the absolute values of the loadings in descending order, and determine which variables the highest loadings correspond to. Figure 4.18 shows that some PCs are loaded predominantly by fewer of the original variables, making them more interpretable, e.g., PC_7, PC_{10}, and PC_{11} are loaded by all four control variables. A possible explanation is that the control variables in this experiment are designed trajectories; hey are predetermined to be variant, and to vary similarly. Hence, they tend to load on the same components, as PCA is derived from the covariance matrix. On the other hand, other principal components are loaded evenly by many variables, making them to interpret.

Since each principal component is generally a combination of all the original variables, it is often nontrivial to have a clear physical interpretation of each PC. This is more evident in problems with a large number of variables that describe distinct physical qualities, as is the case in the model at hand. This limitation is overcome in Sparse PCA as discussed subsequently.



(a) PC 1

(b) PC 2

(c) PC 7

(d) PC 9

(e) PC 10

(f) PC 11

Figure 4.18: PCA loading rank in absolute value for different principal components

4.2.3 Sparse Principal Component Analysis

In this section, the results from the implementation of SPCA on dataset $\tilde{\mathcal{F}}$ are presented. Two sets of sparse principal components are obtained using the two selected approaches discussed in section 2.4. The results from each approach consist of a set of sparse loading vector and the adjusted explained variance of their corresponding sparse PCs.

Formulation A: L_1 penalty (elastic net, lasso)

The first set of sparse loading vectors is computed according to the elastic net approach. The solution is controlled by two parameters. First, λ (eq: 2.18); the ridge regression parameter, which is commonly reserved for cases where the number of variables is larger than the number of samples. However, in this experiment, it was found that having a non-zero λ was beneficial in increasing the explained variance of certain principal components. After λ is set, $\lambda_{1,j}$ (for $j = 1, \ldots, k$; where k is the number of desired PCs), the sparsity-inducing parameters, are determined sequentially for each PC. In this approach, there is no direct relationship between these parameters and the number of non-zero coefficients in the resulting loading vectors, therefore, $\lambda_{1,j}$ were determined experimentally by varying $\lambda_{1,j}$ for the j-th component and choosing a value that produces a loading vector that is sufficiently sparse and produces a PC with a high explained variance. Therefore, the choice of $\lambda_{1,j}$ depends on the user and the requirements of the application.

To choose the parameters in this study, each loading vector (10 in total) was generated using a range of values for λ and λ_1, namely, $0 : 10$ with an increment of 0.5 for λ and $1 : 10$ with an increment of 0.1 for λ_1. For each (λ,λ_1) pair, the sparsity (number of non-zero loadings), and explained variance were recorded. Figure 4.19 shows a plot of these results for the first principal component, displaying how the sparsity and explained variance vary with the two parameters. Based on this, it was decided that it is sufficient to fix λ at any non-zero value, as its effect gradually becomes insignificant for higher values; λ was set to 3.8. On the other hand, λ_1, j were determined after fixing λ with the aid of plots like the one in figure 4.20, making a compromise at with each component between desired sparsity and high explained variance.

Table 4.5 shows the variables corresponding to the non-zero coefficients of the first 10 sparse loading vectors computed with elastic net, along with the loading

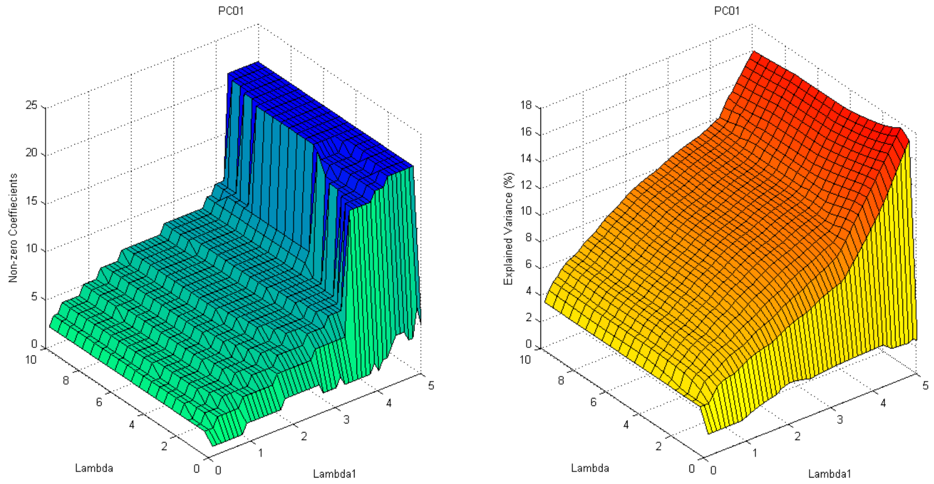


Figure 4.19: The effect of λ and λ_1 on the explained variance (left) and sparsity (right) of the first sparse principal component. The effect of λ diminishes for values slightly larger than zero.

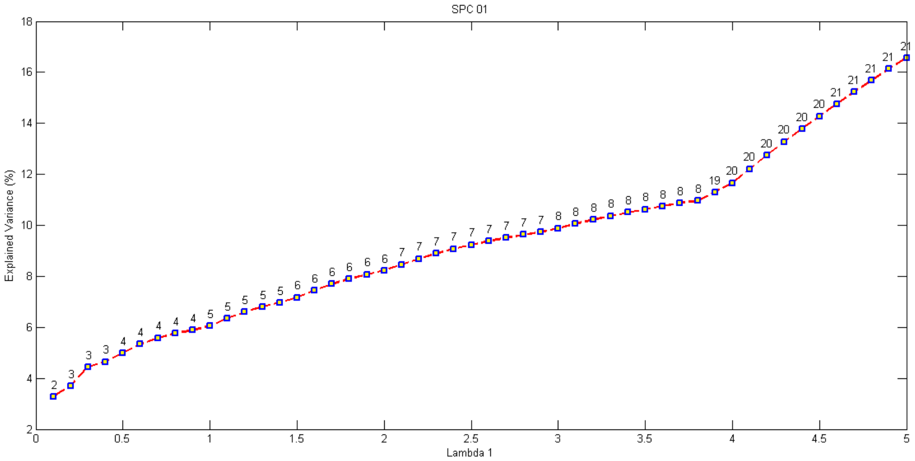


Figure 4.20: The effect of varying λ_1 on the explained variance and sparsity (indicated by the number of non-zero loadings on the plot). The desired sparsity was determined heuristically, then, the value of λ_1 achieving such sparsity and the highest EV possible is selected. In this case, $\lambda_1 = 1.92$ is selected, corresponding to 6 non-zero loadings and 8.22% EV. $\lambda_{1,j}$ was selected similarly for the other sparse PCs.

value corresponding to each variable.

Table 4.6 shows the values of $\lambda_{1,j}$ along with the adjusted explained variance for each sparse component. It is clear that, compared to ordinary PCA, the explained variance of sparse PCs is significantly smaller. For instance, the fist 10 sparse PCs

Table 4.5: Sparse loading vectors using elastic net method; variable name and loading value.

SPC 01		SPC 02		SPC 03		SPC 04		SPC 05	
’Kango_Volume’	0.531	’Kango_Temperature’	-0.508	’TSSmax’	-0.362	’h’	0.592	’SW’	-0.565
’Kango_POPL’	0.242	’Kango_DO’	0.336	’TSS-3’	-0.384	’storage’	0.596	’Cloud-Cover’	0.804
’Kango_POCL’	0.479	’Fukuro_Temperature’	-0.493	’TSS-7’	-0.442	’hTaff’	0.454	’Rain’	0.187
Fukuro_Volume’	0.531	’Fukuro_DO’	0.301	’TSS-13’	-0.458	gateTaff’	-0.296		
’Fukuro_POPL’	0.121	’Air_Temp’	-0.283	’TSS_bot’	-0.413				
’Fukuro_POCL’	0.366	’Taff’	-0.464	’TSS_sed’	-0.380				
SPC 06		**SPC 07**		**SPC 08**		**SPC 09**		**SPC 10**	
’Kango_NH4’	0.065	’u-3’	-0.474	’gateTSSmax’	1.000	’Kango_NH4’	-1.000	’u-7’	-0.757
’Fukuro_NH4’	0.021	’u-7’	-0.478					’u_bot’	0.654
’Wind_Speed’	0.998	’u-13’	-0.526						
		’u_bot’	-0.519						

with this method explain 37% of the total variance, whereas the first ordinary PC explains 42%. On the other hand, the advantage of increased physical interpretability is also clear. The 10 sparse components have at most 6 non-zero loadings, making physical interpretation more plausible. In addition to being sparse, the loading vectors appear to group variables of similar type in a single vector, most notably, loading vectors 4, and 7, which contain total suspended solid, and control variables, respectively; and the first loading vector which contains the inflow volumes, and the POPL, POCL particle concentrations.

Table 4.6: Explained variance and sparsity-inducing parameters $lambda_{1,j}$ for the first 10 sparse PCs.

#	λ_1	sparsity	explained eariance (EV)	cumulative (EV)
1	1.92	6	0.0822	0.0822
2	0.67	6	0.0892	0.1714
3	2.75	6	0.0644	0.2358
4	1.7	4	0.0365	0.2723
5	1.6	3	0.0242	0.2965
6	0.9	3	0.0158	0.3123
7	1.4	4	0.0226	0.3349
8	0.35	1	0.016	0.3509
9	0.1	1	0.0079	0.3588
10	1	2	0.0143	0.3731

Formulation B: L_0 penalty

The second set of sparse principal components were computed using the L_0-norm constrained formulation in 2.1. In this case, the sparsity-inducing parameter s is equal to the number of non-zero coefficients in the resulting loading vector, making the process of selecting the parameters easier.

The algorithm starts with a random initialization of a loading vector (generally not sparse) which converges after some iterations to a sparse loading vector containing s non-zero loadings. Each starting point (or initialization) converges to a different loading vector. Therefore, the experiment is performed by taking 1000 initialization of the first sparse loading vector, and the one that converges to a sparse vector that explaining the highest variance is kept. After the first sparse vector is obtained, deflation is performed with that vector, i.e., its influence is removed from the data matrix (see section 2.4.1), and the process of random initialization is repeated for the second vector, and so on, until all required loading vectors are obtained. Ten loading vectors were obtained using this approach. Results are reported in table 4.7, with their corresponding explained variance in table 4.8.

Table 4.7: Sparse loading vectors with L_0 constraint; variable name and loading value.

SPC 01		**SPC 02**		**SPC 03**		**SPC 04**		**SPC 05**	
’Kango_SSOL1’	0.354	’Kango_Temperature’	0.411	’TSSmax’	0.401	’Kango_NH4’	0.431	’h’	-0.538
’Kango_SSOL3’	0.354	’Kango_DO’	-0.408	’TSS-3’	0.387	’Kango_NO3’	-0.397	’storage’	-0.540
’Kango_SSOL5’	0.354	’Fukuro_Temperature’	0.411	’TSS-7’	0.415	’Kango_PO4’	-0.396	’hTaff’	-0.467
’Kango_SSOL6’	0.354	’Fukuro_DO’	-0.408	’TSS-13’	0.426	’Fukuro_NH4’	0.431	’gateTaff’	0.448
’Fukuro_SSOL1’	0.354	’Air_Temp’	0.406	’TSS_bot’	0.416	’Fukuro_NO3’	-0.397		
’Fukuro_SSOL3’	0.354	’Taff’	0.406	’TSS_sed’	0.404	’Fukuro_PO4’	-0.396		
’Fukuro_SSOL5’	0.354								
’Fukuro_SSOL6’	0.354								

SPC 06		**SPC 07**		**SPC 08**		**SPC 09**		**SPC 10**	
’u-3’	0.495	’T-3’	0.498	’SW’	0.667	’hTSSmax’	0.608	’Wind_Speed’	0.705
’u-7’	0.496	’T-7’	0.502	’Cloud-Cover’	-0.642	’gateTSSmax’	-0.794	’time’	-0.710
’u-13’	0.507	’T-13’	0.502	’Rain’	-0.378				
’u_bot’	0.502	’Tbot’	0.499						

This formulation is characterized by a higher physical interpretability when compared to the previous one, with components 1,3,4,6, and 7 containing variables of the same type. Moreover, the implementation of this formulation is more intuitive, the sparsity controlling parameter is equal to the number of non-zero loadings, making it easy to manipulate the sparsity at each step to achieve a desired effect, such as increased interpretability, or higher explained variance.

It is evident from the implementation of RVS, PCA, and SPCA that a clear dis-

Table 4.8: Explained variance and sparsity for the first 10 sparse PCs using formulation B.

#	sparsity	explained eariance (EV)	cumulative (EV)
1	8	0.089552	0.089552
2	7	0.104477	0.194029
3	6	0.055686	0.249715
4	3	0.022913	0.272628
5	7	0.087912	0.360540
6	2	0.007645	0.368185
7	6	0.056901	0.425086
8	2	0.000288	0.425374
9	3	0.002870	0.428245
10	5	0.003534	0.431779

tinction between the two classes of model-reduction, projection-based and selection-based techniques, is the ability of the latter to select reduced sets of state-variables that have causal relationships with the output or objective of interest, whereas projection-based techniques like PCA and sparse PCA create states based on the maximization of the variance in the input dataset into fewer states; without including any information about the output. Sparse PCA overcomes this problem by creating combination of features that preserve the physical interpretability of the underlying process. In the next section, an attempt to combine projection-based and selection-based approaches is made using the method of weighted principal component analysis; ad-hoc weighting schemes are implemented to achieve that purpose (see section 2.5).

4.2.4 Weighted Principal Component Analysis

In this section, the three weighted principal components analysis schemes (see section 2.5) are applied to dataset $\tilde{\mathcal{F}}$. The first two weighting schemes, which utilize the IIS-ranking of the variables as weights, are expected to have the advantages of both selection-based and projection-based methods. Namely, they should create different sets of PCs, each weighted to suit a particular output of interest, while at the same time achieving the variance compression performance commonly associated with projection-based methods. The third weighting scheme, which uses Pearson's linear correlation coefficient between the inputs and the output to weight the inputs, is proposed as an alternative to the first two schemes; aiming to incorporate the information about the output of interest into the PCs, without the computational burden of IIS.

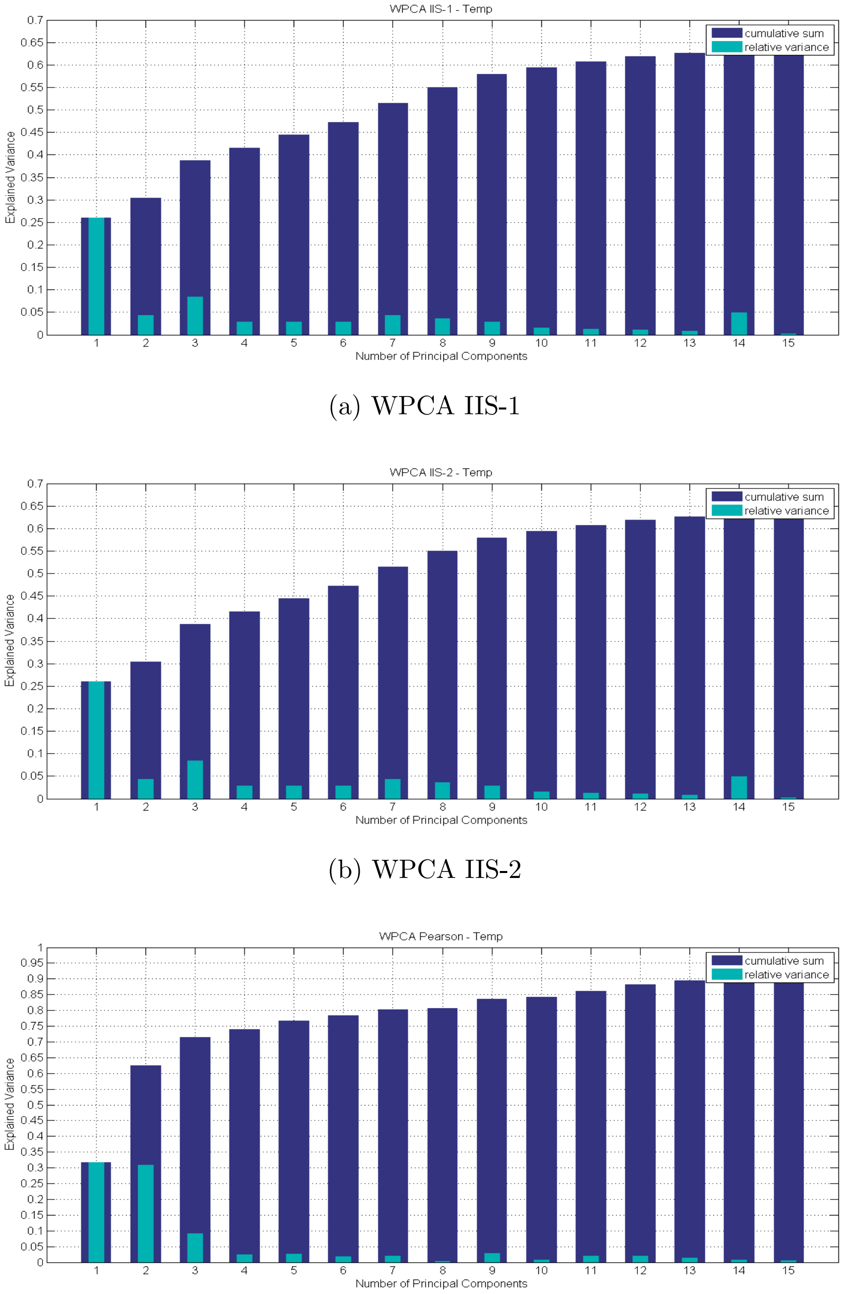
The resulting weighted PCs from the three schemes are analyzed in terms of explained variance and physical interpretability in this section. However, their advantage should become more evident when each set of weighted PCs is emulated with its designated output, in Part II.

Explained Variance

For explained variance, the adjusted explained variance formula will be used (2.4.2), because just as in SCPA, the uncorrelated property of PCA is lost in WPCA, so the correlation between components must be accounted for when computing the variance for a single weighted PC. Another property of ordinary PCA that WPCA does not have is the successive maximization of explained variance, because the foremost loading vectors will tend to be combination of variables with higher weights associated to them, rather than combinations of variables that maximize the total explained variance. Figures 4.23, 4.22, and **??** show the explained variance of the first 15 weighted components using the three weighting schemes on the three outputs: g_{t+1}^{temp}, g_{t+1}^{sed}, and g_{t+1}^{algae}. It can be observed that successive maximization of variance, attributed to ordinary PCs, is not present in weighted PCs; and it appears only partially in some parts, e.g. PCs 12 to 15 in figures **??**, and PCs 1 to 3 from the third weighted scheme (**??**, **??**, and **??**).

Physical Interpretability

Figure 4.24 shows the heat maps of the loading vectors from WPCA using all weighting schemes on g_{t+1}^{temp}, which demonstrates a side-effect of WPCA: the loadings vectors appear more *sparse* when compared to ordinary PCA. This sparsity is not complete as many loadings are close to zero but not exactly equal to zero, however, it still improves the overall interpretability of the PCs. This sparsity is a direct result of emphasizing and de-emphasizing, a priori, different input variables using the weighting schemes, and the higher the relative difference in weights among the variables, the more sparse the loading vectors tend to be, e.g. the loading vectors from the second weighting scheme are more sparse than those from the first and third schemes because the first scheme weights the variables based on a logarithmic scale, as opposed to a linear scale.

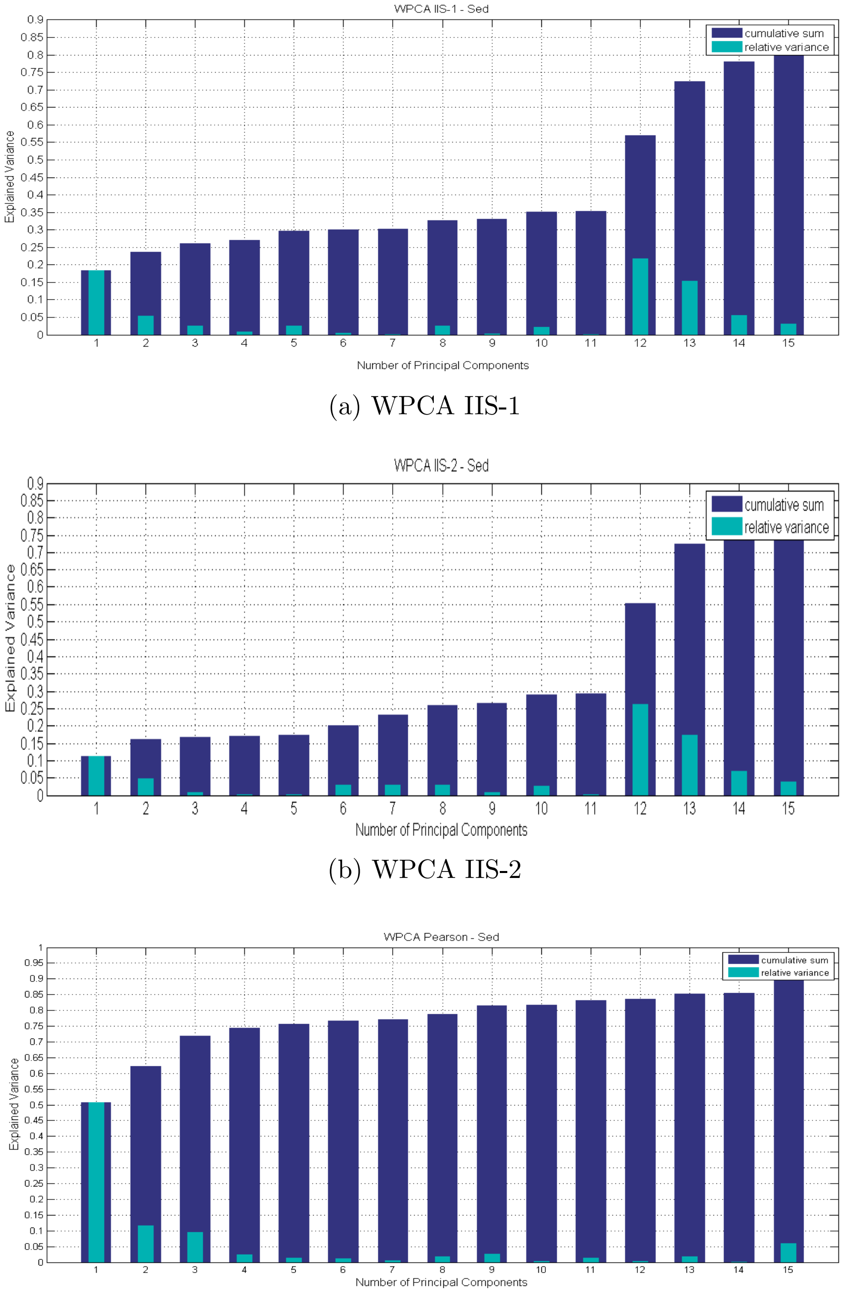


(a) WPCA IIS-1

(b) WPCA IIS-2

(c) WPCA Pearson

Figure 4.21: Adjusted explained variance and cumulative adjusted explained variance for weighted PCs from the three schemes on g_{t+1}^{temp}.

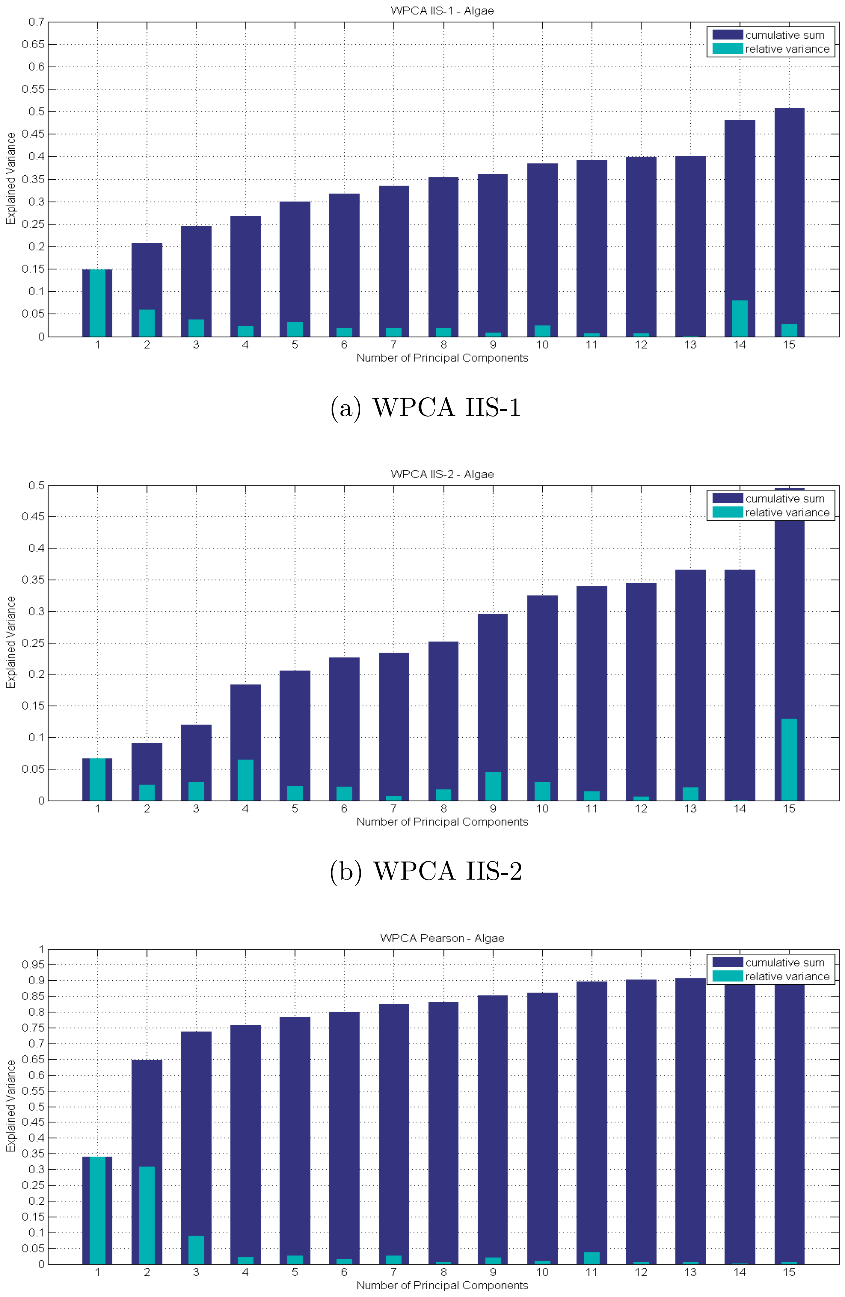


(a) WPCA IIS-1

(b) WPCA IIS-2

(c) WPCA Pearson

Figure 4.22: Adjusted explained variance and cumulative adjusted explained variance for weighted PCs from the three schemes on g_{t+1}^{sed}.

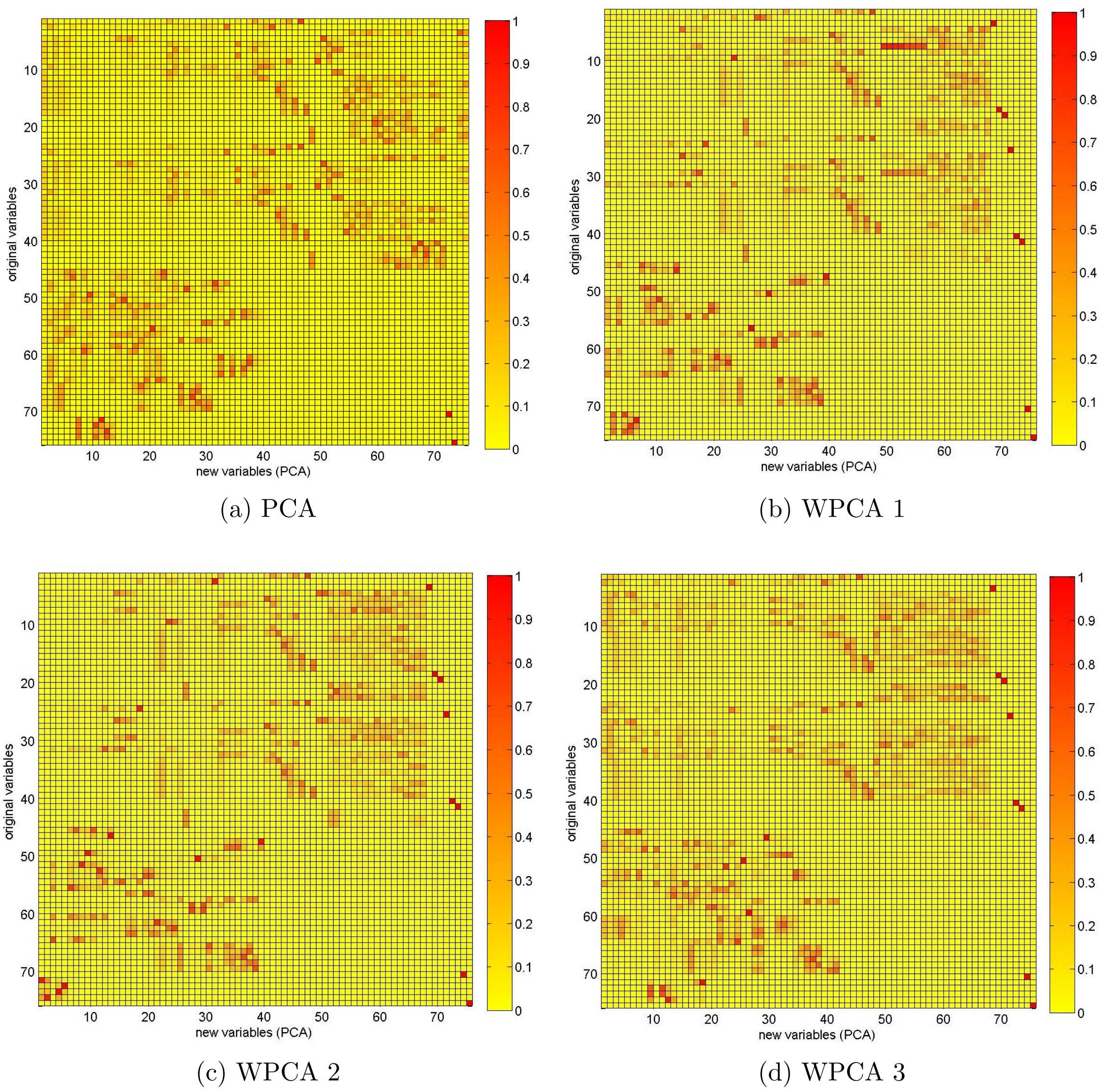


(a) WPCA IIS-1

(b) WPCA IIS-2

(c) WPCA Pearson

Figure 4.23: Adjusted explained variance and cumulative adjusted explained variance for weighted PCs from the three schemes on g_{t+1}^{algae}.



(a) PCA

(b) WPCA 1

(c) WPCA 2

(d) WPCA 3

Figure 4.24: Loading vector heat maps for (a)ordinary PCA, (b)WPCA with RVS rank 1, (c)WPCA with RVS rank 2, (d)WPCA with Pearson coefficients.

4.3 Part III: Model identification and comparison

The next step in the dynamic emulation modeling exercise is to identify the dynamic emulator models for the different outputs of interest using the sets of reduced variables selected in Part II of this chapter. The model structure that will be used is Extra-Trees, the same structure used by the IIS algorithm. The Extra-Trees models are parameterized similarly to the ranking experiments, namely: M, the number of trees in the ensemble is set to 100, n_{min}, the minimum cardinality for splitting a node is 2, while K, the number of alternative cut directions corresponds to the number of inputs in the emulator (Geurts et al., 2006). The emulators are validated with k-fold cross validation (with $k = 10$).

4.3.1 Emulation results

Recursive Variable Selection (RVS)

The emulator performances for the selected causal networks in tables 4.2, 4.3, and 4.4 are reported in tables 4.9, 4.10, 4.11, respectively. The emulators are built only with the variables selected for the output, i.e., the first iteration of RVS, and the performance is shown in terms of R^2 and RMSE.

Table 4.9: Emulator performance for selected causal networks for g_{t+1}^{temp}

minimum δR^2 (%)	Number of inputs (states, controls, exogenous)	Extra-Trees performance	
		R^2	RMSE
0	44 (19, 4, 21)	0.831039	0.407382
0.5	23 (8, 4, 11)	0.830576	0.407504
1	20 (7, 4, 9)	0.825909	0.412604
1.5	18 (6, 4, 8)	0.818915	0.420671
2	16 (6, 4, 6)	0.800926	0.441164
2.5	15 (6, 4, 5)	0.800926	0.441164
3	14 (5, 4, 5)	0.800926	0.441164
3.5	14 (5, 4, 5)	0.800926	0.441164
4	9 (2, 4, 3)	0.738681	0.506655
4.5	9 (2, 4, 3)	0.738681	0.506655
5	9 (2, 4, 3)	0.738681	0.506655

As expected, the performance of the emulators decreases with the number of inputs. Furthermore, outputs g_{t+1}^{sed} (total released sediments) and g_{t+1}^{algae} (algal bloom) achieve better accuracy than g_{t+1}^{temp} (temperature) for similarly sized networks, e.g

Table 4.10: Emulator performance for selected causal networks for g_{t+1}^{sed}

minimum δR^2 (%)	Number of inputs (states, controls, exogenous)	Extra-Trees performance	
		R^2	RMSE
0	4 (1,1,2)	0.957969	0.0811967
0.5	4 (1,1,2)	0.957969	0.0811967
1	3 (1, 0, 2)	0.957969	0.0811967
1.5	3 (1, 0, 2)	0.957969	0.0811967
2	3 (1, 0, 2)	0.957969	0.0811967
2.5	3 (1, 0, 2)	0.957969	0.0811967
3	3 (1, 0, 2)	0.957969	0.0811967
3.5	3 (1, 0, 2)	0.957969	0.0811967
4	3 (1, 0, 2)	0.957969	0.0811967
4.5	3 (1, 0, 2)	0.957969	0.0811967
5	3 (1, 0, 2)	0.957969	0.0811967

with 2 state-variables, g_{t+1}^{temp} and g_{t+1}^{algae} achieve $R^2 = 0.738$ and 0.928 respectively, while g_{t+1}^{sed} achieves $R^2 = 0.958$ with a single state-variable. This is an indicator that the dynamics of temperature are more complex than algal bloom or sediments. Nonetheless, even with two state-variables only, the obtained g_{t+1}^{temp} emulator is considered accurate enough for use in designing the optimal operating policy for the selective withdrawal system of Tono Dam. An emulator for g_{t+1}^{temp} with matching performance has been successfully used for that purpose by Castelletti et al. (2012b).

Pincipal Component Analysis (PCA, SPCA, and WPCA)

The performances of dynamic emulators built with the different sets of principal components are reported here. First, the performance in terms of R^2 of emulators built with ordinary PCs are compared to those built with the weighted PCs to highlight the effect of the different weighting schemes. Figures 4.25, 4.26, and 4.27 show the R^2 scores of emulators as a function of the number of used PCs (or weighted PCs).

For g_{t+1}^{temp} (fig. 4.25), the performance increases significantly after the inclusion of the seventh component in ordinary PCA, which contains high loadings from the control variables (see fig. 4.18:c). This is consistent with the RVS results, which showed the importance of the control variables in explaining g_{t+1}^{temp}. However, for WPCA, the emulator performance starts increasing at the the inclusion of the second component, due to the fact that the important variables for the output were prompted to load on the foremost components by the weighting scheme. As the

Table 4.11: Emulator performance for selected causal networks for g_{t+1}^{algae}

minimum δR^2 (%)	Number of inputs (states, controls, exogenous)	Extra-Trees performance	
		R^2	RMSE
0	43 (19, 4, 20)	0.951804	0.219428
0.5	25 (10, 4, 11)	0.948452	0.22693
1	11 (3, 4, 4)	0.934896	0.255047
1.5	11 (3, 4, 4)	0.934896	0.255047
2	11 (3, 4, 4)	0.934896	0.255047
2.5	11 (3, 4, 4)	0.934896	0.255047
3	9 (2, 4, 3)	0.928012	0.268198
3.5	9 (2, 4, 3)	0.928012	0.268198
4	9 (2, 4, 3)	0.928012	0.268198
4.5	9 (2, 4, 3)	0.928012	0.268198
5	9 (2, 4, 3)	0.928012	0.268198

number of components increases, the WPCA emulators outperform the PCA ones by about 10%.

For g_{t+1}^{sed}, all emulators, from PCA and WPCA, reach 90% accuracy with the inclusion of the first principal component only. This can be attributed to the fact that most of g_{t+1}^{sed}'s dynamic behavior can be explained by a number of exogenous drivers related to suspended solids (see 4.2.1) which load on the first PC (see fig. 4.18:a). Similarly, for g_{t+1}^{algae}, the weighting schemes do not a have a big impact. In fact, emulators from ordinary PCA and all WPCA schemes reach 90% with only 5 components.

In the case of sparse PCA, the principal components were ranked using IIS to determine their importance in explaining each of the outputs. This was done because there is no sequential maximization of variance in sparse PCA and the sparseness is chosen by the user, and hence, no natural order of the PCs exists to aid in choosing among them for emulation. Tables 4.12, 4.15, and 4.18 show the performance of the emulator built with sparse PCs from both considered formulations. The first column refers to the number of PCs in the emulator, while the second column refers to the number associated to the sparse component added to the emulator; taken from tables 4.5 and 4.7, for formulations A and B respectively.

The results show that sparse PCs can achieve similar accuracy to ordinary PCs, despite having lower explained variance, as was shown in Part I of this chapter. Indeed, they are easier to interpret physically. For instance, an emulator for g_{t+1}^{sed} built with the first two selected sparse PCs of formulation B has an R^2 score of

0.91; and it is composed of sparse PCs 4 and 1 from table 4.7, which can be easily interpreted, as they collectively contain variables that describe concentrations of different particles.

Table 4.12: SPCA emulation performance for g_{t+1}^{temp}

Table 4.13: Formulation A: elastic-net

#	sPC	R^2
1	2	0.183155
2	4	0.322982
3	7	0.426112
4	3	0.487195
5	9	0.541715
6	6	0.553051
7	1	0.563721
8	5	0.570645
9	10	0.572179

Table 4.14: Formulation B: L_0-constrained

#	sPC	R^2
1	2	0.185676
2	5	0.310413
3	6	0.414242
4	3	0.478696
5	4	0.531768
6	10	0.572015
7	7	0.59484
8	1	0.60473
9	9	0.610344

Table 4.15: SPCA emulation performance for g_{t+1}^{sed}

Table 4.16: Formulation A: elastic-net

#	sPC	R^2
1	1	0.903198
2	3	0.953258
3	7	0.962269
4	6	0.964415
5	5	0.966304
6	9	0.967128
7	10	0.967848
8	2	0.968434
9	4	0.968501

Table 4.17: Formulation B: L_0-constrained

#	sPC	R^2
1	4	0.868694
2	1	0.915863
3	3	0.953592
4	6	0.963922
5	5	0.966627
6	10	0.967521

Table 4.18: SPCA emulation performance for g_{t+1}^{algae}

Table 4.19: Formulation A: elastic-net

#	sPC	R^2
1	2	0.44434
2	1	0.637168
3	4	0.805177
4	3	0.866776
5	6	0.884869
6	9	0.899786
7	5	0.909181
8	7	0.909415
9	8	0.909595

Table 4.20: Formulation B: L_0-constrained

#	sPC	R^2
1	2	0.445296
2	4	0.637323
3	5	0.799247
4	3	0.862036
5	7	0.886097
6	10	0.905278
7	8	0.914188
8	6	0.914269
9	9	0.915088

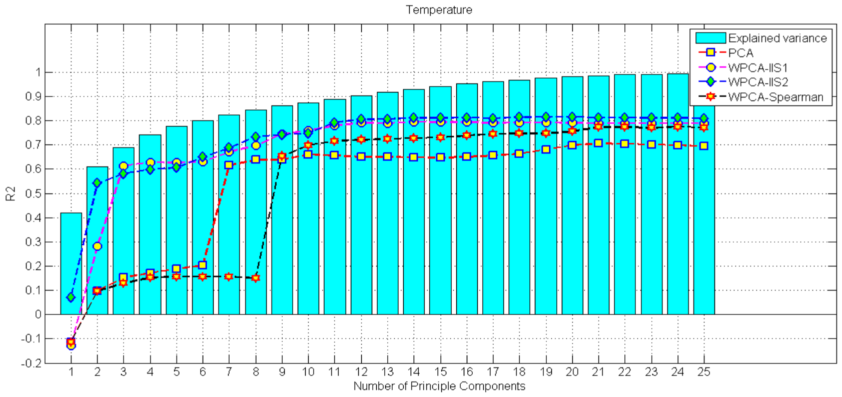


Figure 4.25: comparison of the PCA explained variance with respect to the PCA and WPCA emulator performance of g_{t+1}^{temp} for increasing number of principal components.

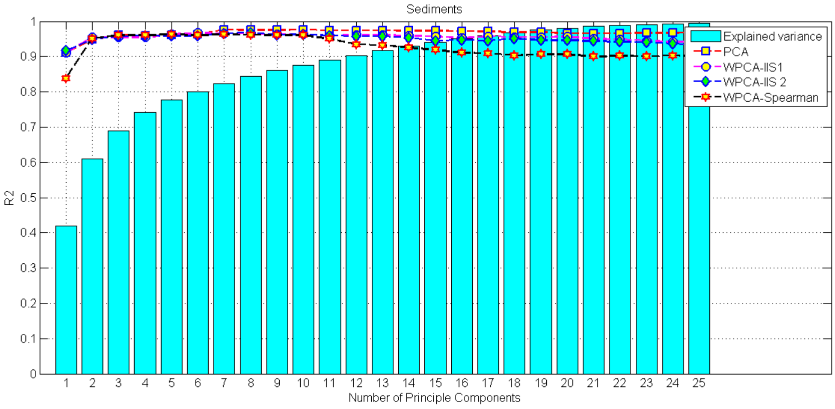


Figure 4.26: comparison of the PCA explained variance with respect to the PCA and WPCA emulator performance of g_{t+1}^{sed} for increasing number of principal components.

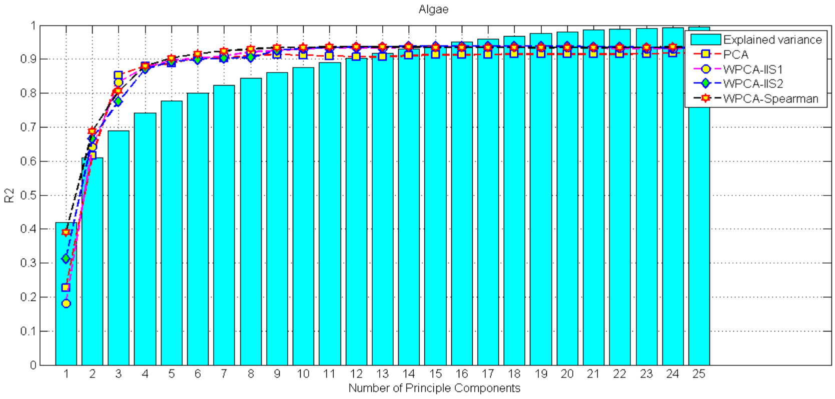


Figure 4.27: comparison of the PCA explained variance with respect to the PCA and WPCA emulator performance of g_{t+1}^{algae} for increasing number of principal components.

4.3.2 Evaluation and Comparison

Finally, the different model-reduction techniques are compared using a few similarly sized sets of reduced variables from each technique, and the performance of their corresponding emulator models in terms of R^2 is compared. The size or the network is defined as the number of state-variables in the selection-based emulators, and the number of principal components in projection-based emulators.

In order to select the best principal components for each emulator output, IIS was applied on the set of ordinary principal components, and the two sets of sparse principal components. Then, the fist k components from the ranking are used to build the emulator, where k refers to the size of the network to be analyzed. This way, the best ordinary PCs are compared to the best sparse PCs in explaining a certain output, making the comparison objective. On the contrary, the first k weighted PCs were selected based on their order due to the fact that a ranking (RVS or Spearman correlation) was incorporated a priori into the components by the weighting schemes.

Table 4.21 shows that for temperature, WPCA with the first weighting scheme achieves the highest accuracy among projection-based approaches, using 3, 5, and 10 components from each approach. When similarly sized networks from RVS are compared to the projection-based approaches, similar or better accuracy is observed, keeping in mind that these networks contain other inputs that are not state-variables; the total number of variables is reported between brackets in the table.

For released sediments and algal bloom, there is less apparent variability in performance among the different approaches. Interestingly, SPCA achieves slightly higher accuracy than ordinary PCA in all three output variables. This gives it another advantage besides being easier to interpret, and demonstrates that higher explained variance in PCA methods does not guarantee higher accuracy in emulation exercises. Overall, all techniques were successful in achieving high fidelity while reducing the number of state-variables of the original model; with RVS and SPCA having the extra benefit of physical interpretability.

Table 4.21: Emulator performance comparison for g_{t+1}^{temp}

Reduction method	**3 PCs**	**5 PCs**	**10 PCs**
	R^2	R^2	R^2
PCA	0.218805	0.49399	0.568526
SPCA (elastic-net)	0.426112	0.541715	0.572179
SPCA (L_0 contraint)	0.414242	0.531768	0.610344
WPCA (RVS-1)	0.613296	0.627268	0.759034
WPCA (RVS-2)	0.581056	0.606571	0.74819
WPCA (Spearman coeff.)	0.128939	0.156599	0.698822
Reduction method	**2 states (9)**	**5 states (14)**	**8 states (23)**
	R^2	R^2	R^2
RVS	0.738681	0.800926	0.831039

Table 4.22: Emulator performance comparison for g_{t+1}^{sed}

Reduction method	**3 PCs**	**5 PCs**	**10 PCs**
	R^2	R^2	R^2
PCA	0.878365	0.890362	0.959705
SPCA (elastic-net)	0.903198	0.953258	0.964415
SPCA (L_0 contraint)	0.868694	0.915863	0.963922
WPCA (RVS-1)	0.910707	0.956024	0.954615
WPCA (RVS-2)	0.916525	0.949493	0.95929
WPCA (Spearman coeff.)	0.838868	0.951421	0.96112
Reduction method	**1 state (4 variables)**		
	R^2		
RVS	0.957969		

Table 4.23: Emulator performance comparison for g_{t+1}^{algae}

Reduction method	**1 PCs**	**2 PCs**	**4 PCs**
	R^2	R^2	R^2
PCA	0.70979	0.786227	0.885419
SPCA (elastic-net)	0.805177	0.884869	0.909595
SPCA (L_0 contraint)	0.799247	0.886097	0.915088
WPCA (RVS-1)	0.832015	0.897011	0.930044
WPCA (RVS-2)	0.775622	0.892947	0.931553
WPCA (Spearman coeff.)	0.805639	0.903584	0.935002
Reduction method	**2 states (9)**	**3 states (11)**	**10 states (25)**
	R^2	R^2	R^2
RVS	0.928012	0.934896	0.948452

Conclusions

In this work, an exhaustive experimental comparison was made between two classes of model-order reduction approaches; projection-based and selection-based. The comparison was performed on the reduction of DYRESM-CAEDYM, a 1D hydro-ecological model used to describe the in-reservoir water quality conditions of Tono Dam, an artificial reservoir located in western Japan.

Projection-based approaches, like the popular Principal Component Analysis (PCA), have been used extensively to create reduced-order models (emulators) of complex process-based models. Despite being an effective and efficient solution, the working principle of these techniques, which consists of using snapshots of the original model to project the state-variables onto a lower dimensional space, creates the disadvantage of including all the original state-variables in the emulator. In this study, a selection-based model-reduction technique called Recursive Variable Selection (RVS) (Castelletti et al., 2012b) was used as an alternative to projection-based approaches. RVS uses the information contained in the snapshots to select the state variables of the original model that are relevant with respect to the emulator's output and discards irrelevant ones, thus reducing complexity.

Experiments on three output-variables of the Tono Dam system (water temperature, released sediments, and Chlorophyll-a concentration) reveal that the states selected by RVS can be used to build emulators with higher accuracy than those built with principal components, while maintaining a lower number of state-variables in the emulator. In addition to high accuracy, RVS emulators were easy to interpret and they often revealed physically meaningful relationships between the outputs and the states.

Morever, two additional techniques, sparse PCA and weighted PCA were applied to the case study to mitigate the drawbacks of ordinary PCA. In sparse PCA, sparse loading vectors were obtained from the snapshots, and were used to create sparse principal components, which are linear combination of only a few of the original states. The emulators built with sparse PCs achieved higher accuracy than ordinary PCs, while the sparsity caused the resulting emulators to be easier to interpret physically.

In weighted PCA, an ad-hoc weighting scheme was developed to emphasize input variables that have a causal relationship with the output of interest prior to performing PCA. The resulting weighted components were used to build emulators that outperformed ordinary and sparse PCs in R^2 accuracy, specially for the water temperature output.

Overall, the results from the projection-based approaches; PCA, sparse PCA, and weighted PCA, revealed that choosing components with higher explained variance does not guarantee better emulator accuracy, as the high variance components are not necessarily relevant to the output of interest.

Future research includes utilizing the emulators obtained from the competing approaches in designing the optimal operating policy of the reservoir, and verifying if the emulators with higher accuracy produce better policies.

Lastly, due to the good performance of sparse PCA in this study, it would be beneficial to develop a formal approach of constructing sparse components, and choosing sparsity based on expert-based knowledge of the outputs that the components will be used to emulate, so as to improve on the results obtained from the heuristic approach used in this study.

Appendix A

DEMo procedure and algorithms

A.1 Variables involved in the DEMo procedure

- $\mathcal{M}$, original process-based model.
- $\mathbf{X}_t, \mathbf{Y}_t, \mathbf{W}_t$, physically-based model state, output and exogenous driver vector.
- $\mathbf{x}_t, \mathbf{y}_t, \mathbf{w}_t, \mathbf{u}_t$ emulator state, output, exogenous driver, and control vectors.
- $\mathcal{N}_x, \mathcal{N}_y, \mathcal{N}_w, \mathcal{N}_u$, dimensionality of the state, output, exogenous driver, and control vectors respectively.
- $f_t(\cdot), h_t(\cdot)$, emulator state transition and output transformation functions.
- $\tilde{\mathbf{X}}_t, \tilde{\mathbf{W}}_t$, physically-based model state and exogenous driver vectors (after aggregation).
- $\mathcal{F}$, data-set of tuples $\{\mathbf{X}_t, \mathbf{W}_t, \mathbf{u}_t, \mathbf{Y}_t, \mathbf{X}_{t+1}\}(witht = 1, \ldots, H$ foe the DEMo process.
- $\tilde{\mathcal{F}}$, data-set of tuples $\{\tilde{\mathbf{X}}_t, \tilde{\mathbf{W}}_t, \mathbf{u}_t, \mathbf{Y}_t, \tilde{\mathbf{X}}_{t+1}\}(witht = 1, \ldots, H)$ foe the DEMo process (after aggregation).
- H, simulation horizon.

A.2 Variables involved in the RVS-IIS algorithms

- $\mathbf{v}_t^i = \{\tilde{\mathbf{X}}_t, \tilde{\mathbf{W}}_t, \mathbf{u}_t\}, \mathbf{v}_t^o = \{\tilde{\mathbf{X}}_{t+1}, \mathbf{Y}_t\}$ input and output data, respectively, employed in the variable selection process.

- v_t^o, i-th component of the vector $\mathbf{Y}_t$.

- $v^i = \{\tilde{\mathbf{X}}_t, \tilde{\mathbf{W}}_t, \mathbf{u}_t\}, v^o = \{\tilde{\mathbf{X}}_{t+1} + \mathbf{Y}_t\}$, set of the candidate input and output variables for the variable selection process.

- v_{tar}^i, subset of the output variables that need to be explained ($v_{tar}^i \subseteq v^o$).

- v_{sel}^i, set of the input variables selected during the i-th iteration of the variable selection process.

- $v_{v^o}^i$, set of the input variables that will appear in the output transformation function for explaining v^o.

- $v_{\tilde{\mathbf{X}}_t}^n ew = v_{v^o}^i \cap \tilde{\mathbf{X}}_t$, set of the output variables to be explained.

- $v_{\mathbf{Y}}^i$, set of the input variables to explain the output $\mathbf{Y}$.

- v^*, most significant variable added to the set $v_{v^o}^i$.

- $\hat{m}(\cdot)$, underlying model to explain v^o.

- $\hat{v}^o$, residuals of $\hat{m}(\cdot)$.

- $D(v^o, \hat{m}(v_{v^o}^i))$, distance metric between the output v^o and the model $\hat{m}(\cdot)$ predictions.

A.3 Recursive Variable Selection algorithm

Algorithm 1 RVS($\tilde{\mathcal{F}}$, $\mathcal{V}^o_{tar}$, $\mathcal{V}^i_{sel}$): Recursive Variable Selection

```
Require: The dataset $\tilde{\mathcal{F}}$, the set $\mathcal{V}^o_{tar}$ of variables to be explained and the set $\mathcal{V}^i_{sel}$
    of previously selected variables
Ensure: $\mathcal{V}^i_{\mathcal{V}^o_{tar}}$: the set of input variables to explain $\mathcal{V}^o_{tar}$
  Initialize: $\mathcal{V}^i_{\mathcal{V}^o_{tar}} \leftarrow \emptyset$
  //For each variable that has to be explained
  for all $v^o \in \mathcal{V}^o_{tar}$ do
    //Select, with a suitable IS algorithm, the most relevant variables to explain $v^o$
    $\mathcal{V}^i_{v^o} \leftarrow \mathrm{IS}(\tilde{\mathcal{F}}, v^o)$
    //Consider the new state variables, i.e. not yet in $\mathcal{V}^i_{sel}$
    $\mathcal{V}^{new}_{\tilde{\mathbf{X}}} \leftarrow (\mathcal{V}^i_{v^o} \setminus \mathcal{V}^i_{sel}) \cap \tilde{\mathbf{X}}$
    //Add variables obtained by recursively execute RVS
    $\mathcal{V}^i_{v^o} \leftarrow \mathcal{V}^i_{v^o} \cup \mathrm{RVS}(\tilde{\mathcal{F}}, \mathcal{V}^{new}_{\tilde{\mathbf{X}}}, \mathcal{V}^i_{sel} \cup \mathcal{V}^i_{v^o})$
    //Add the selected input variables $\mathcal{V}^i_{v^o}$ to the set of input variables to be returned
    $\mathcal{V}^i_{\mathcal{V}^o_{tar}} \leftarrow \mathcal{V}^i_{\mathcal{V}^o_{tar}} \cup \mathcal{V}^i_{v^o}$
  end for
  return $\mathcal{V}^i_{\mathcal{V}^o_{tar}}$
```

A.4 Iterative Input Selection algorithm

Algorithm 2 IIS($\tilde{\mathcal{F}}$, v^o): Iterative Input Selection

```
Require: The dataset $\tilde{\mathcal{F}}$ and the variable $v^o$ to be explained
Ensure: $\mathcal{V}^i_{v^o}$: the set of variables selected to explain $v^o$
  Initialize: $\mathcal{V}^i_{v^o} \leftarrow \emptyset$, $\hat{v}^o \leftarrow v^o$, $D_{old} \leftarrow 0$
  repeat
    //With an Input Ranking (IR) algorithm, select the most relevant input variable $v^*$ to explain $\hat{v}^o$
    $v^* \leftarrow \mathrm{IR}(\tilde{\mathcal{F}}, \hat{v}^o)$
    //If such variable has been previously selected, then the algorithm stops and returns the set $\mathcal{V}^i_{v^o}$ of the input variables selected up to that point
    if $v^* \in \mathcal{V}^i_{v^o}$ then
      return $\mathcal{V}^i_{v^o}$
    end if
    //Add $v^*$ to the set $\mathcal{V}^i_{v^o}$ of selected variables
    $\mathcal{V}^i_{v^o} \leftarrow \mathcal{V}^i_{v^o} \cup v^*$
    //By using $\tilde{\mathcal{F}}$ estimate a model $\hat{m}(\cdot)$ that explains the variable $v^o$ using $\mathcal{V}^i_{v^o}$ as argument
    $\hat{m}(\cdot) \leftarrow \mathrm{MB}(\tilde{\mathcal{F}}, v^o, \mathcal{V}^i_{v^o})$
    //Compute the residuals
    $\hat{v}^o \leftarrow v^o - \hat{m}(\mathcal{V}^i_{v^o})$
    //Compute the variation of the coefficient of determination
    $\Delta D \leftarrow D(v^o, \hat{m}(\mathcal{V}^i_{v^o})) - D_{old}$
    //Backup $D$ for the next iteration
    $D_{old} \leftarrow D(v^o, \hat{m}(\mathcal{V}^i_{v^o}))$
    //Stop iterating when the improvement is too low
  until $\Delta D < \epsilon$
  return $\mathcal{V}^i_{v^o}$
```

Appendix B

Iterative Input Selection Results for RVS

The IIS results were obtained using the following parameters for Extra-trees: $K = 75$, $n_{min} = 2$, $M = 500$ for outputs and 100 for state-variables.

Taxonomy

- **#:** the number of the selected variables as listed in table 4.1.
- **variable name:** the name or abbreviated name of the variable, see (Hipsey et al., 2006) for details on the variables.
- **rel. δR^2 (%):** the relative R^2 contribution of the selected variable to the built model, normalized by the total R^2 score of the model built by all selected variables (in percent).
- **abs. δR^2:** the absolute value of R^2 contribution of the selected variable to the built model.
- **abs δR^2 (cum.):** the absolute value of R^2 score of the model built by all variables up to the current variable.

Table B.1: IIS for g^{temp}

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
55	hTaff	3.75082	0.03032	0.03032
64	Tsed	5.10450	0.04126	0.07157
24	Fukuro_Temperature	8.37528	0.06769	0.13927
73	u-13	6.98426	0.05645	0.19572
72	u-7	12.39428	0.10018	0.29590
74	u_bot	20.00186	0.16167	0.45756
71	u-3	29.11105	0.23529	0.69286
52	h	9.10573	0.07360	0.76646
60	T-3	1.44199	0.01166	0.77811
51	time	1.77505	0.01435	0.79246
45	SW	0.54203	0.00438	0.79684
54	Taff	0.04479	0.00036	0.79720
49	Wind_Speed	0.18299	0.00148	0.79868
9	Kango_DO	0.43093	0.00348	0.80216
31	Fukuro_DO	0.21008	0.00170	0.80386
46	Cloud-Cover	0.11432	0.00092	0.80478
61	T-7	0.18930	0.00153	0.80631
58	hTSSmax	0.24076	0.00195	0.80826

Table B.2: IIS for g^{sed}

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
11	Kango_POCL	94.36316	0.91272	0.91272
65	TSS-3	5.63684	0.05452	0.96724

Table B.3: IIS for g^{algae}

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
51	time	55.38861	0.52030	0.52030
4	Kango_NH4	27.88374	0.26193	0.78223
52	h	10.06631	0.09456	0.87679
65	TSS-3	2.71227	0.02548	0.90227
64	Tsed	0.65726	0.00617	0.90844
49	Wind_Speed	0.79458	0.00746	0.91590
61	T-7	0.70867	0.00666	0.92256
31	Fukuro_DO	0.93372	0.00877	0.93133
60	T-3	0.24038	0.00226	0.93359
55	hTaff	0.22260	0.00209	0.93568
47	Air_Temp	0.39186	0.00368	0.93936

Table B.4: IIS for Taff(54)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
54	Taff	97.70370	0.96810	0.96810
45	SW	0.49180	0.00487	0.97297
47	Air_Temp	1.24701	0.01236	0.98533
55	hTaff	0.17914	0.00178	0.98710
49	Wind_Speed	0.10335	0.00102	0.98813
46	Cloud-Cover	0.05803	0.00058	0.98870
52	h	0.02402	0.00024	0.98894
60	T-3	0.03603	0.00036	0.98930
64	Tsed	0.01181	0.00012	0.98942
9	Kango_DO	0.09628	0.00095	0.99037
53	storage	0.00252	0.00003	0.99039
31	Fukuro_DO	0.03694	0.00037	0.99076
50	Rain	0.00262	0.00003	0.99079
72	u-7	0.00676	0.00007	0.99085

Table B.5: IIS for hTaff(55)

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
55	hTaff	57.11129	0.47111	0.47111
51	time	10.85559	0.08955	0.56065
47	Air_Temp	9.06093	0.07474	0.63540
46	Cloud-Cover	3.87638	0.03198	0.66737
60	T-3	7.32665	0.06044	0.72781
50	Rain	2.24211	0.01850	0.74631
49	Wind_Speed	1.86958	0.01542	0.76173
53	storage	1.20452	0.00994	0.77166
64	Tsed	0.99989	0.00825	0.77991
48	Vap_Press	1.76399	0.01455	0.79446
54	Taff	0.91733	0.00757	0.80203
24	Fukuro_Temperature	1.34417	0.01109	0.81312
61	T-7	0.42163	0.00348	0.81660
74	u_bot	0.17954	0.00148	0.81808
62	T-13	0.30210	0.00249	0.82057
69	TSS_sed	0.52431	0.00433	0.82489

Table B.6: IIS gateTaff(56)

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
56	gateTaff	48.00651	0.31671	0.31671
52	h	10.43650	0.06885	0.38556
45	SW	3.86874	0.02552	0.41109
63	Tbot	5.38817	0.03555	0.44663
47	Air_Temp	13.91690	0.09181	0.53845
55	hTaff	1.96764	0.01298	0.55143
64	Tsed	1.07469	0.00709	0.55852
49	Wind_Speed	2.29308	0.01513	0.57364
68	TSS_bot	1.61825	0.01068	0.58432
62	T-13	1.90049	0.01254	0.59686
46	Cloud-Cover	1.39998	0.00924	0.60609
74	u_bot	0.40047	0.00264	0.60874
72	u-7	0.26663	0.00176	0.61050
60	T-3	1.22309	0.00807	0.61856
54	Taff	0.65800	0.00434	0.62291
73	u-13	0.19993	0.00132	0.62422
51	time	1.14154	0.00753	0.63176
66	TSS-7	0.42109	0.00278	0.63453
61	T-7	0.29361	0.00194	0.63647
57	TSSmax	0.36682	0.00242	0.63889
69	TSS_sed	0.16765	0.00111	0.64000
65	TSS-3	0.17598	0.00116	0.64116
48	Vap_Press	0.84793	0.00559	0.64675
50	Rain	0.18023	0.00119	0.64794
59	gateTSSmax	0.04805	0.00032	0.64826
31	Fukuro_DO	0.57949	0.00382	0.65208
9	Kango_DO	0.25662	0.00169	0.65377
24	Fukuro_Temperature	0.32286	0.00213	0.65590
71	u-3	0.09231	0.00061	0.65651
7	Kango_PO4	0.25086	0.00166	0.65817
58	hTSSmax	0.06048	0.00040	0.65857
6	Kango_PONL	0.17538	0.00116	0.65972

Table B.7: IIS for TSSmax(57)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
12	Kango_SSOL1	76.04237449	0.65242	0.65242
50	Rain	17.24234792	0.147934	0.800354
47	Air_Temp	6.715277592	0.057615	0.857969

Table B.8: IIS for hTSSmax(58)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
58	hTSSmax	43.59824176	0.322157	0.322157
45	SW	3.903118326	0.028841	0.350998
51	time	21.80609049	0.16113	0.512128
47	Air_Temp	16.90584392	0.124921	0.637049
67	TSS-13	4.353774823	0.032171	0.66922
48	Vap_Press	3.935868739	0.029083	0.698303
68	TSS_bot	1.143963774	0.008453	0.706756
49	Wind_Speed	3.037939052	0.022448	0.729204
64	Tsed	0.464054393	0.003429	0.732633
66	TSS-7	0.851104717	0.006289	0.738922

Table B.9: IIS for gateTSSmax(59)

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
59	gateTSSmax	23.98262257	0.0934881	0.0934881
53	storage	8.015807458	0.0312469	0.124735
73	u-13	5.781189074	0.022536	0.147271
71	u-3	3.80435898	0.01483	0.162101
58	hTSSmax	1.955537997	0.007623	0.169724
47	Air_Temp	8.575071316	0.033427	0.203151
74	u_bot	2.461674226	0.009596	0.212747
45	SW	4.71761036	0.01839	0.231137
49	Wind_Speed	2.940105075	0.011461	0.242598
72	u-7	2.816970058	0.010981	0.253579
67	TSS-13	3.025786525	0.011795	0.265374
63	Tbot	5.659080181	0.02206	0.287434
55	hTaff	3.153795637	0.012294	0.299728
50	Rain	2.914965009	0.011363	0.311091
52	h	1.999148316	0.007793	0.318884
65	TSS-3	1.149516695	0.004481	0.323365
64	Tsed	1.178248199	0.004593	0.327958
54	Taff	1.869600016	0.007288	0.335246
51	time	1.650009235	0.006432	0.341678
46	Cloud-Cover	1.029716584	0.004014	0.345692
60	T-3	1.233145894	0.004807	0.350499
48	Vap_Press	1.40579145	0.00548	0.355979
7	Kango_PO4	5.555441542	0.021656	0.377635
61	T-7	0.335542923	0.001308	0.378943
62	T-13	0.440464219	0.001717	0.38066
31	Fukuro_DO	0.478943912	0.001867	0.382527
9	Kango_DO	0.77959858	0.003039	0.385566
2	Kango_Temperature	0.184445995	0.000719	0.386285
24	Fukuro_Temperature	0.308094075	0.001201	0.387486
56	gateTaff	0.193681121	0.000755	0.388241
28	Fukuro_PONL	0.404036776	0.001575	0.389816

Table B.10: IIS for T-3(60)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
60	T-3	86.6167326	0.800239	0.800239
52	h	4.130059477	0.038157	0.838396
71	u-3	1.697830358	0.015686	0.854082
73	u-13	1.002072769	0.009258	0.86334
54	Taff	2.672410527	0.02469	0.88803
4	Kango_NH4	1.333715776	0.012322	0.900352
74	u_bot	0.971657728	0.008977	0.909329
72	u-7	0.986919368	0.009118	0.918447
48	Vap_Press	0.136272372	0.001259	0.919706
45	SW	0.1478539	0.001366	0.921072
47	Air_Temp	0.066566726	0.000615	0.921687
50	Rain	0.13031925	0.001204	0.922891
49	Wind_Speed	0.024245442	0.000224	0.923115
2	Kango_Temperature	0.03961532	0.000366	0.923481
55	hTaff	0.017101696	0.000158	0.923639
53	storage	0.017967604	0.000166	0.923805
57	TSSmax	0.008659086	8.00E-05	0.923885

Table B.11: IIS for T-7(61)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
60	T-3	87.13337357	0.79409	0.79409
52	h	2.811214133	0.02562	0.81971
71	u-3	1.835189554	0.016725	0.836435
54	Taff	2.752290558	0.025083	0.861518
73	u-13	1.318264114	0.012014	0.873532
72	u-7	0.982937401	0.008958	0.88249
74	u_bot	1.01541669	0.009254	0.891744
4	Kango_NH4	1.635815	0.014908	0.906652
56	gateTaff	0.149668075	0.001364	0.908016
48	Vap_Press	0.122894607	0.00112	0.909136
45	SW	0.151533439	0.001381	0.910517
55	hTaff	0.091402864	0.000833	0.91135

Table B.12: IIS for T-13(62)

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
62	T-13	86.64558712	0.780156	0.780156
52	h	3.246893877	0.029235	0.809391
71	u-3	2.079633585	0.018725	0.828116
72	u-7	1.019548	0.00918	0.837296
2	Kango_Temperature	3.487898143	0.031405	0.868701
73	u-13	1.256331915	0.011312	0.880013
74	u_bot	1.160818704	0.010452	0.890465
50	Rain	0.405375839	0.00365	0.894115
55	hTaff	0.145491055	0.00131	0.895425
45	SW	0.1861397	0.001676	0.897101
48	Vap_Press	0.141826013	0.001277	0.898378
31	Fukuro_DO	0.08596189	0.000774	0.899152
53	storage	0.138494156	0.001247	0.900399

Table B.13: IIS for Tbot(63)

#	variable name	rel. δR^2 (%)$	abs. δR^2	abs δR^2 (cum.)
63	Tbot	86.29188505	0.782544	0.782544
52	h	3.009625553	0.027293	0.809837
71	u-3	2.045636743	0.018551	0.828388
73	u-13	1.191918902	0.010809	0.839197
54	Taff	3.135444728	0.028434	0.867631
72	u-7	1.042170927	0.009451	0.877082
74	u_bot	1.141194257	0.010349	0.887431
26	Fukuro_NH4	1.685491759	0.015285	0.902716
48	Vap_Press	0.089650298	0.000813	0.903529
45	SW	0.173787047	0.001576	0.905105
50	Rain	0.174448673	0.001582	0.906687
55	hTaff	0.018746065	0.00017	0.906857

Table B.14: IIS for Tsed(64)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
64	Tsed	90.53285269	0.832283	0.832283
71	u-3	0.92220738	0.008478	0.840761
55	hTaff	1.706812456	0.015691	0.856452
73	u-13	0.643630699	0.005917	0.862369
51	time	1.630777665	0.014992	0.877361
72	u-7	0.578473561	0.005318	0.882679
52	h	1.595425294	0.014667	0.897346
74	u_bot	0.689643169	0.00634	0.903686
48	Vap_Press	0.273246631	0.002512	0.906198
50	Rain	0.476658733	0.004382	0.91058
9	Kango_DO	0.14869751	0.001367	0.911947
4	Kango_NH4	0.734785427	0.006755	0.918702
54	Taff	0.066788786	0.000614	0.919316

Table B.15: IIS for TSS-3(65)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
14	Kango_SSOL3	83.98453	0.68545	0.68545
65	TSS-3	15.16894	0.12380	0.80925
71	u-3	0.84652	0.00691	0.81616

Table B.16: IIS for TSS-7(66)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
14	Kango_SSOL3	80.6622812	0.694278	0.694278
65	TSS-3	12.71037571	0.109401	0.803679
45	SW	6.627343091	0.057043	0.860722

Table B.17: IIS for TSS-13(67)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
39	Fukuro_SSOL6	90.14684046	0.682975	0.682975
65	TSS-3	3.694043887	0.027987	0.710962
45	SW	5.518561294	0.04181	0.752772
46	Cloud-Cover	0.640554364	0.004853	0.757625

Table B.18: IIS for TSSbot(68)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
39	Fukuro_SSOL6	74.73786639	0.67743	0.67743
50	Rain	17.04133238	0.154464	0.831894
51	time	8.220801229	0.074514	0.906408

Table B.19: IIS for TSSsed(69)

#	variable name	rel. δR^2 (%)\$	abs. δR^2	abs δR^2 (cum.)
14	Kango_SSOL3	71.87626556	0.653131	0.653131
50	Rain	18.5566443	0.168622	0.821753
51	time	9.567090134	0.086935	0.908688

Bibliography

Alaa H Aly and Richard C Peralta. Optimal design of aquifer cleanup systems under uncertainty using a neural network and a genetic algorithm. *Water Resources Research*, 35(8):2523–2532, 1999.

E. Audsley, K.R. Pearn, P.A. Harrison, and P.M. Berry. The impact of future socio-economic and climate changes on agricultural land use and the wider environment in east anglia and north west england using a metamodel system. *Climatic Change*, 90(1-2):57–88, 2008. ISSN 0165-0009. doi: 10.1007/s10584-008-9450-9. URL `http://dx.doi.org/10.1007/s10584-008-9450-9`.

Alexandre M Baltar and Darrell G Fontane. Use of multiobjective particle swarm optimization in water resources management. *Journal of Water Resources Planning and Management*, 134(3):257–265, 2008.

J Bartholow, RB Hanna, L Saito, D Lieberman, and M Horn. Simulated limnological effects of the shasta lake temperature control device. *Environmental Management*, 27(4):609–626, 2001.

Rajib Kumar Bhattacharjya and Bithin Datta. Optimal management of coastal aquifers using linked simulation optimization approach. *Water resources management*, 19(3):295–320, 2005.

Christen Duus Børgesen, Jørgen Djurhuus, and Arne Kyllingsbæk. Estimating the effect of legislation on nitrogen leaching by upscaling field simulations. *Ecological Modelling*, 136(1):31–48, 2001.

Aziz Bouzaher, PG Lakshminarayan, Richard Cabe, Alicia Carriquiry, Philip W Gassman, and Jason F Shogren. Metamodels and nonpoint pollution policy in agriculture. *Water Resources Research*, 29(6):1579–1587, 1993.

DR Broad, Graeme Clyde Dandy, and Holger R Maier. Water distribution system optimization using metamodels. *Journal of Water Resources Planning and Management*, 131(3):172–180, 2005.

J. Cadima and I. Jolliffe. Loadings and correlations in the interpretation of principal components. *Journal of Applied Statistics*, 22:203–214, 1995.

A Castelletti, S Galelli, and R Soncini-Sessa. A tree-based feature ranking approach to enhance emulation modelling of 3d hydrodynamic-ecological models. *International Environmental Modelling and Software Society, Ottawa, Ont., Canada*, pages 5–8, 2010a.

A Castelletti, F Pianosi, R Soncini-Sessa, and JP Antenucci. A multiobjective response surface approach for improved water quality planning in lakes and reservoirs. *Water Resources Research*, 46(6), 2010b.

A Castelletti, S Galelli, M Restelli, and R Soncini-Sessa. Tree-based feature selection for dimensionality reduction of large-scale control systems. In *Proceedings of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, Paris, France*, 2011.

A Castelletti, S Galelli, M Ratto, R Soncini-Sessa, and PC Young. A general framework for dynamic emulation modelling in environmental problems. *Environmental Modelling & Software*, 34:5–18, 2012a.

A Castelletti, S Galelli, M Restelli, and R Soncini-Sessa. Data-driven dynamic emulation modelling for the optimal management of environmental systems. *Environmental Modelling & Software*, 34:30–43, 2012b.

A. Castelletti, H. Yajima, M. Giuliani, R. Soncini-Sessa, and E. Weber. Planning the optimal operation of a multioutlet water reservoir with water quality and quantity targets. *Journal of Water Resources Planning and Management*, 140 (4):496–510, 2014. doi: 10.1061/(ASCE)WR.1943-5452.0000348. URL `http://dx.doi.org/10.1061/(ASCE)WR.1943-5452.0000348`.

Andrea Castelletti, AV Lotov, and Rodolfo Soncini-Sessa. Visualization-based multiobjective improvement of environmental decision-making using linearization of response surfaces. *Environmental Modelling & Software*, 25(12):1552–1564, 2010c.

Raymond B Cattell. The scree test for the number of factors. *Multivariate behavioral research*.

Qiuming Cheng, Greame Bonham-Carter, Wenlei Wang, Shengyuan Zhang, Wenchang Li, and Xia Qinglin. A spatially weighted principal component analysis for multi-element geochemical data for mapping locations of felsic intrusions in the gejiu mineral district of yunnan, china. *Computers & Geosciences*, 37(5):662–669, 2011.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

R Douglas Evans. Empirical evidence of the importance of sediment resuspension in lakes. *Hydrobiologia*, 284(1):5–12, 1994.

Darrell G Fontane, John W Labadie, and Bruce Loftis. Optimal control of reservoir discharge quality through selective withdrawal. *Water Resources Research*, 17(6): 1594–1602, 1981.

S Galelli. *Dealing with complexity and dimensionality in water resources management*. PhD thesis, PhD thesis, Politec. di Milano, Milan, Italy, 2010.

S Galelli and A Castelletti. Tree-based iterative input variable selection for hydrological modeling. *Water Resources Research*, 49(7):4295–4310, 2013.

S Galelli, C Gandolfi, R Soncini-Sessa, and D Agostani. Building a metamodel of an irrigation district distributed-parameter model. *Agricultural water management*, 97(2):187–200, 2010.

R.K. Gelda and S.W. Effler. Modeling turbidity in a water supply reservoir: Advancements and issues. *Journal of Water Resources Planning and Management*, 2:139–148, 2007.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

M Giuliani, S Galelli, and R Soncini-Sessa. A dimensionality reduction approach for many-objective markov decision processes: Application to a water reservoir operation problem. *Environmental Modelling & Software*, 57:101–114, 2014.

Matteo Giuliani. Dealing with multi criteria problems in water resources planning and management. 2010.

Graham Clifford Goodwin and Robert L Payne. Dynamic system identification: experiment design and data analysis. 1977.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and L Zadeh. Feature extraction. *Foundations and applications*, 2006.

Tsuyoshi Hashimoto, Jery R Stedinger, and Daniel P Loucks. Reliability, resiliency, and vulnerability criteria for water resource system performance evaluation. *Water resources research*, 18(1):14–20, 1982.

M.R. Hipsey, J.R. Romero, J.P. Antenucci, and D. Hamilton. *Computational Aquatic Ecosystem Dynamics Model: CAEDYM v2.3 Science Manual.* Centre for Water Research, University of Western Australia, 2006a.

MR Hipsey, JR Romero, JP Antenucci, and D Hamilton. Computational aquatic ecosystem dynamics model: Caedym v2. *Contract Research Group, Centre for Water Research, University of Western Australia*, page 90, 2006b.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psycology*, 24:417–441, 1933.

A. Imerito. *Dynamic Reservoir Simulation Model: DYRESM Science Manual.* Centre for Water Research, University of Western Australia, 2007.

Virginia M Johnson and Leah L Rogers. Accuracy of neural network approximators in simulation-optimization. *Journal of Water Resources Planning and Management*, 126(2):48–56, 2000.

Ian Jolliffe. *Principal component analysis.* Wiley Online Library, 2005.

Kees Jong, Jérémie Mary, Antoine Cornuéjols, Elena Marchiori, and Michèle Sebag. Ensemble feature ranking. In *Knowledge Discovery in Databases: PKDD 2004*, pages 267–278. Springer, 2004.

Henry F Kaiser. The application of electronic computers to factor analysis. *Educational and psychological measurement*, 1960.

Kari Karhunen. *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*, volume 37. Universitat Helsinki, 1947.

Patrick M Kocovsky, Jean V Adams, and Charles R Bronte. The effect of sample size on the stability of principal components analysis of truss-based fish morphometrics. *Transactions of the American Fisheries Society*, 138(3):487–496, 2009.

Valentina Krysanova and Uwe Haberlandt. Assessment of nitrogen leaching from arable land in large river basins: Part i. simulation experiments using a process-based model. *Ecological Modelling*, 150(3):255–275, 2002.

Bimlesh Kumar, Gopu Sreenivasulu, and Achanta Ramakrishna Rao. Metamodel-based design of alluvial channels at incipient motion subjected to seepage. *Hydrological Sciences Journal–Journal des Sciences Hydrologiques*, 55(3):459–466, 2010.

John A Lee and Michel Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.

CM Liaw, Ching-Tsai Pan, and M Ouyang. Model reduction of discrete systems using the power decomposition method and the system identification method. In *Control Theory and Applications, IEE Proceedings D*, volume 133, pages 30–34. IET, 1986.

Lester W Mackey. Deflation methods for sparse pca. In *Advances in neural information processing systems*, pages 1017–1024, 2009.

Lin Mei, Michael Figl, Daniel Rueckert, Ara Darzi, and Philip Edwards. Statistical shape modelling: How many modes should be retained? In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2008.

Deyu Meng, Qian Zhao, and Zongben Xu. Improve robustness of sparse pca by¡ i¿ l¡/i¿¡ sub¿ 1¡/sub¿-norm maximization. *Pattern Recognition*, 45(1):487–497, 2012.

D.F. Morrison. *Metodi di analisi statistica multivariata*. Casa Editrice Ambrosiana, Milano, I, 1976.

TR Neelakantan and NV Pundarikanthan. Neural network-based simulation-optimization model for reservoir operation. *Journal of water resources planning and management*, 126(2):57–64, 2000.

Jason W Osborne and Anna B Costello. Sample size and subject to item ratio in principal components analysis. *Practical Assessment, Research & Evaluation*, 9 (11):8, 2004.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

Elazar J Pedhazur. Multiple regression in behavioral research: Explanation and prediction. 1997.

JD Piñeros Garcet, A Ordonez, J Roosen, and Marnik Vanclooster. Metamodelling: Theory, concepts and application to nitrate leaching modelling. *Ecological Modelling*, 193(3):629–644, 2006.

Joaquim F Pinto da Costa, Hugo Alonso, and Luis Roque. A weighted principal component analysis and its application to gene expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 8(1):246–252, 2011.

Nestor V Queipo, Raphael T Haftka, Wei Shyy, Tushar Goel, Rajkumar Vaidyanathan, and P Kevin Tucker. Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1):1–28, 2005.

Patrick M Reed and Joshua B Kollat. Visual analytics clarify the scalability and effectiveness of massively parallel many-objective optimization: A groundwater monitoring design example. *Advances in Water Resources*, 56:1–13, 2013.

Peter Richtárik, Martin Takáč, and Selin Damla Ahipaşaoğlu. Alternating maximization: unifying framework for 8 sparse pca formulations and efficient parallel codes. *arXiv preprint arXiv:1212.4137*, 2012.

Leah L Rogers and Farid U Dowla. Optimization of groundwater remediation using artificial neural networks with parallel solute transport modeling. *Water Resources Research*, 30(2):457–481, 1994.

S. Sadocchi. *Manuale di analisi statistica multivariata.* Editore Franco Angeli, Milano, I, 1990.

Andrea Saltelli, Karen Chan, E Marian Scott, et al. *Sensitivity analysis*, volume 134. Wiley New York, 2000.

Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

Karl Sjöstrand, Line Harder Clemmensen, Rasmus Larsen, and Bjarne Ersbøll. Spasm: A matlab toolbox for sparse statistical modeling. *Journal of Statistical Software Accepted for publication*, 2012.

DP Solomatine and A Torres. Neural network approximation of a hydrodynamic model in optimizing reservoir operation. In *Proceeding of the Second International Conference on Hydroinformatics, Zurich, Switzerland*, 1996.

R. Soncini-Sessa, A. Castelletti, and E. Weber. *Integrated and participatory water resources management: Theory.* Elsevier, Amsterdam, NL, 2007a.

J Sreekanth and Bithin Datta. Multi-objective management of saltwater intrusion in coastal aquifers using genetic programming and modular neural network based surrogate models. *Journal of Hydrology*, 393(3):245–256, 2010.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, Series B 58:267–288, 1996.

Rudolph van der Merwe, Todd K Leen, Zhengdong Lu, Sergey Frolov, and Antonio M Baptista. Fast neural network surrogates for very high dimensional physics-based models in computational oceanography. *Neural Networks*, 20(4):462–478, 2007.

S. Vines. Simple Principal Components. *Applied Statistics*, 49:441–451, 2000.

Warren M Washington, Lawrence Buja, and Anthony Craig. The computational future for climate and earth system models: on the path to petaflop and beyond. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1890):833–846, 2009.

Louis A Wehenkel. *Automatic learning techniques in power systems.* Number 429. Springer, 1998.

A.J. Jakeman Wheater and K.J. Beven. Modelling change in environmentalsystems. 1993.

Karen Willcox and Jaime Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330, 2002.

Corey Winton, Jackie Pettway, CT Kelley, Stacy Howington, and Owen J Eslinger. Application of proper orthogonal decomposition (pod) to inverse problems in saturated groundwater flow. *Advances in Water Resources*, 34(12):1519–1526, 2011.

M Xu, PJ Van Overloop, and NC Van De Giesen. Model reduction in model predictive control of combined water quantity and quality in open channels. *Environmental Modelling & Software*, 42:72–87, 2013.

H. Yajima, S. Kikkawa, and J. Ishiguro. Effect of selective withdrawal system operation on the long-and short-term water conservation in a reservoir. *Annual Journal of Hydraulic Engineering, JSCE*, 50:1375–1380 (in Japanese), 2006.

Shengquan Yan and Barbara Minsker. A dynamic meta-model approach to genetic algorithm solution of a risk-based groundwater remediation design model. *Bridges*, 10(40685):99, 2003.

PC Young and Marco Ratto. A unified approach to environmental systems modeling. *Stochastic Environmental Research and Risk Assessment*, 23(7):1037–1057, 2009.

Peter Young. Data-based mechanistic modelling, generalised sensitivity and dominant mode analysis. *Computer Physics Communications*, 117(1):113–129, 1999.

H Henry Yue and Masayuki Tomoyasu. Weighted principal component analysis and its applications to improve fdc performance. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 4, pages 4262–4267. IEEE, 2004.

Weihua Zhang and Bernd Michaelis. Shape control with karhunen-loève-decomposition: Theory and experimental results. *Journal of intelligent material systems and structures*, 14(7):415–422, 2003.

H. Zou, T. Hastie, and R. Tibshirani. Sparse Principal Component Analysis. *Journal of Computational and Graphical Statistics.*, 2004.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2): 301–320, 2005.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.