Technische Universität Braunschweig

Master's Thesis

Global Sensitivity Analysis for Vector-Valued Responses of Mechanical Models

Prateek Bhustali

September 14, 2021

Institute for Dynamics and Vibrations
Prof. Dr.-Ing. Ulrich Römer
Institute of Structural Analysis
Prof. Dr.-Ing. Ursula Kowalsky

Supervisor:
Julius Schultz, M.Sc.

Technische Universität Braunschweig

Institut für IDS Dynamik und Schwingungen

Technische Universität Braunschweig | Institut für Dynamik und Schwingungen
Schleinitzstr. 20 | 38106 Braunschweig | Deutschland

Mr. Prateek Bhustali
4941936

Technische Universität Braunschweig
Institut für Dynamik und Schwingungen

Schleinitzstraße 20, 2. OG
38106 Braunschweig
Deutschland

Prof. Dr.-Ing. Ulrich Römer

Fon. +49 (0) 531 391-62120
Fax +49 (0) 531 391-7017
u.roemer@tu-braunschweig.de
www.ids.tu-bs.de

Datum: 15. März 2021

Task Sheet for the M.Sc. Thesis
Global Sensitivity Analysis for Vector-Valued Responses of Mechanical Models
Globale Sensitivitätsanalyse für vektorwertige Ausgangsgrößen von mechanischen Modellen

The field of global sensitivity analysis has received considerable attention in recent years, in part, because it allows to quantify sensitivities in the presence of strong nonlinearities and parameter dependencies. Global sensitivities employ stochastic analysis and relate output variability to (combined) input variability, where variability is most commonly expressed through the variance. Sobol' coefficients are the most prominent example of a variance-based global sensitivity analysis. Currently, the task of computing such sensitivities for vector-valued responses is investigated in some detail. Discrete time-responses and finite element models have vector-valued responses and computing sensitivities for each component individually may be inefficient. Also, defining a single global quantity may not always be desired or possible. Hence, dedicated coefficients, which are intrinsically vector-valued, need to be defined and computed.

The thesis will be concerned with global sensitivity methods for vector-valued responses and their efficient computation. In particular, the following points need to be addressed:

1. Literature research; getting familiar with global sensitivity analysis, for vector-valued functions
2. Implement generalized Sobol coefficients for vector valued functions
    a. Getting familiar with Karhunen-Loève expansion code
    b. Getting familiar with Polynomial Chaos code
    c. Combine both to obtain an efficient implementation for the generalized sensitivity indices
    d. Numerical tests on toy example (harmonic oscillator)
3. Apply the framework to a mechanical example
    a. Starting point is an elastic model, which should be extended to include a simple Chaboche model

  b. Realistic modeling of uncertain input distributions
  c. Sensitivity analysis for vector-valued response (discussion, interpretation of results and computational efficiency)
  d. Apply single-output sensitivity analysis and discuss the advantages and short-comings
4. Optional: Realize a vector-valued Chatterjee sensitivity indices as a generalization
  a. Recover Sobol indices through post-processing
  b. Numerical tests on toy example → Sobol Function / Ishigami Function
  c. Improve sampling based approach of generalized Sobol indices

Basic references:

1. Alexanderian, Alen, Pierre A. Gremaud, and Ralph C. Smith. "Variance-based sensitivity analysis for time-dependent processes." *Reliability Engineering & System Safety* 196 (2020): 106722.
2. Gamboa, Fabrice, et al. "Global Sensitivity Analysis: a new generation of mighty estimators based on rank statistics." *arXiv preprint arXiv:2003.01772* (2020).

Remark (Reference 2 is technical, the proofs are not important for the thesis)

Thesis duration: 15.03.2021-15.09.2021

1st Examiner

Prof. Dr.-Ing. Ulrich Römer U. Röme

2nd Examiner

Prof. Dr.-Ing. Ursula Kowalsky

Supervisor

M.Sc. Julius Schultz

Statement of Originality

This thesis has been performed independently with the support of my supervisor/s. To the best of the author's knowledge, this thesis contains no material previously published or written by another person except where due reference is made in the text.

Braunschweig, September 14, 2021

Abstract

Mathematical models are approximations of reality which enable us predict the outcome of natural phenomena. However, the inputs to such models may not be known perfectly which results in output uncertainty. Sensitivity analysis aims to quantify the contributions of the stochastic inputs towards the output uncertainty. In this work, we focus on global sensitivity analysis of time-dependent models and use generalised Sobol indices, an extension of the classical variance-based Sobol indices, as a metric to compute their sensitivities. This method is applied to the viscoplastic response of the Chaboche model, to quantify the contribution of the material parameters towards the stress response. In order to expedite computations, two surrogate modeling techniques are employed. Finally, the insights gathered from the sensitivity studies are used to carry out a basic Bayesian calibration of the Young's modulus.

Keywords: Global Sensitivity Analysis, Sobol Indices, Time-dependent processes, Chaboche Model, Surrogate modeling, Karnhunen-Loève Expansion, Polynomial Chaos, Bayesian Calibration, Uncertainty Quantification

Preface

Here we present a short summary of each section to help navigate through this work efficiently:

Chapter 1 is dedicated to reviewing some ideas from probability theory, owing to their central role in Uncertainty Quantification. The ideas presented here are a summary of the self-contained textbook "Probability Theory and Stochastic Processes" by Pierre Brémaud [11] and the lecture notes "Methods of Uncertainty Analysis and Quantification" by Prof. Ulrich Römer [23]. For illustration, we introduce the example of a mechanical oscillator which we shall reference throughout this work.

Chapter 2 introduces two surrogate modeling techniques, Polynomial Chaos Expansions (PCE) and Karnhunen-Loève Expansion with PCE approximated modes (KLE+PCE). For most problems of practical importance, the computational model is time and resource intensive and becomes a bottleneck for UQ studies. Therefore, such models are replaced by computationally cheaper approximations called surrogates.

Chapter 3 covers an important area in Uncertainty Quantification called Sensitivity Analysis, which is main focus of this work. We introduce the classical Sobol indices, a variance-based sensitivity metric and their extension to time-dependent processes, generalised Sobol indices. We also illustrate the use of surrogate models from chapter 2 for their efficient computation.

Chapter 4 introduces the Chaboche model, the application example in this work. The model has been summarised with a focus on practical implementation and the model response has been illustrated for two settings, monotonic loading and cyclic loading.

In Chapter 5, we present the results of this work. We begin by building surrogates, introduced in Chapter 2, for the Chaboche model, which is quintessential to circumventing the computational expense. We compute Sobol indices and generalised Sobol indices of the Chaboche model using the surrogates. Using the insights from the sensitivity analysis, we carry out a basic Bayesian calibration to calibrate the Young's modulus using the model and the surrogates and discuss the results.

In Chapter 6, we conclude this thesis and provide an outlook for future work.

(The surrogate models, sensitivity indices and the Chaboche model have been implemented in Python and can downloaded from the GitLab respository)

Contents

List of Figures

List of Tables

1. Introduction

1.1. Uncertainty Quantification

Mathematical models allow us to describe our understanding of physical phenomena and processes in the language of mathematics and are an indispensable part of study in many fields. These models allow us to circumvent physical experiments to predict the outcome(s) of any experiment, which are often time-consuming and expensive [37, p. 1].

However, these models only capture a part of reality and often contain simplifications for mathematical and computational convenience. Therefore, predictions about the real world using such models contain inherent uncertainties. *Uncertainty Quantification*(UQ) deals with the quantification and alleviation of uncertainties arising from such models.

Uncertainties can be classified into two types, *aleatory* and *epistemic* uncertainty. The word aleatory stems from the Latin word *alea* which means 'a dice game' and refers to the inherent "randomness" in a process, which is irreducible. The word *epistemic* is derived from the Greek word *epistēmē* which means 'knowledge' and refers to uncertainties arising from our incomplete understanding or from a deliberate simplification of a process. Such kind of uncertainties can be eliminated by ameliorating our understanding of the process and are therefore reducible [31].

In order to elucidate these ideas, let us consider a spring-mass-damper system as in Fig.1.1 under the influence of a force $F \in \mathbb{R}$, where the position of the mass block 'm' is governed by a second order ordinary differential equation

$$m\ddot{x} + c\dot{x} + kx = F, \quad x \in \mathbb{R}. \tag{1.1}$$

In the deterministic setting, the parameters, mass (m), spring stiffness (k) and damping coefficient (c) are modeled as positive constants. However, in practice, these parameters are not constants. For example, the springs produced by a machine do not have identical stiffness and there are inherent/random differences in the value. This is an example of aleatory uncertainty which cannot be reduced and must be accounted to predict the position of the block. Similarly, the ODE (1.1) does not consider additional physical effects such as friction, drag etc. which influence the system in the real world. Uncertainties arising from such simplifications are termed epistemic uncertainties and we must incur a computational cost to reduce them due to the complex system of equations arising thereof.

We shall now only focus on the above mentioned aleatory uncertainties on the position of the block. The histogram of the values of the parameters when measured experimentally could, for example, look like the orange (k), green (c) and black (m) plots in Figure 1.1, where the x-axis represents the value of the parameter and the y-axis represents the frequency of that value. We can now solve the ODE (1.1) using different values of the parameters and plot the position of the block at some time $t = t_1$.

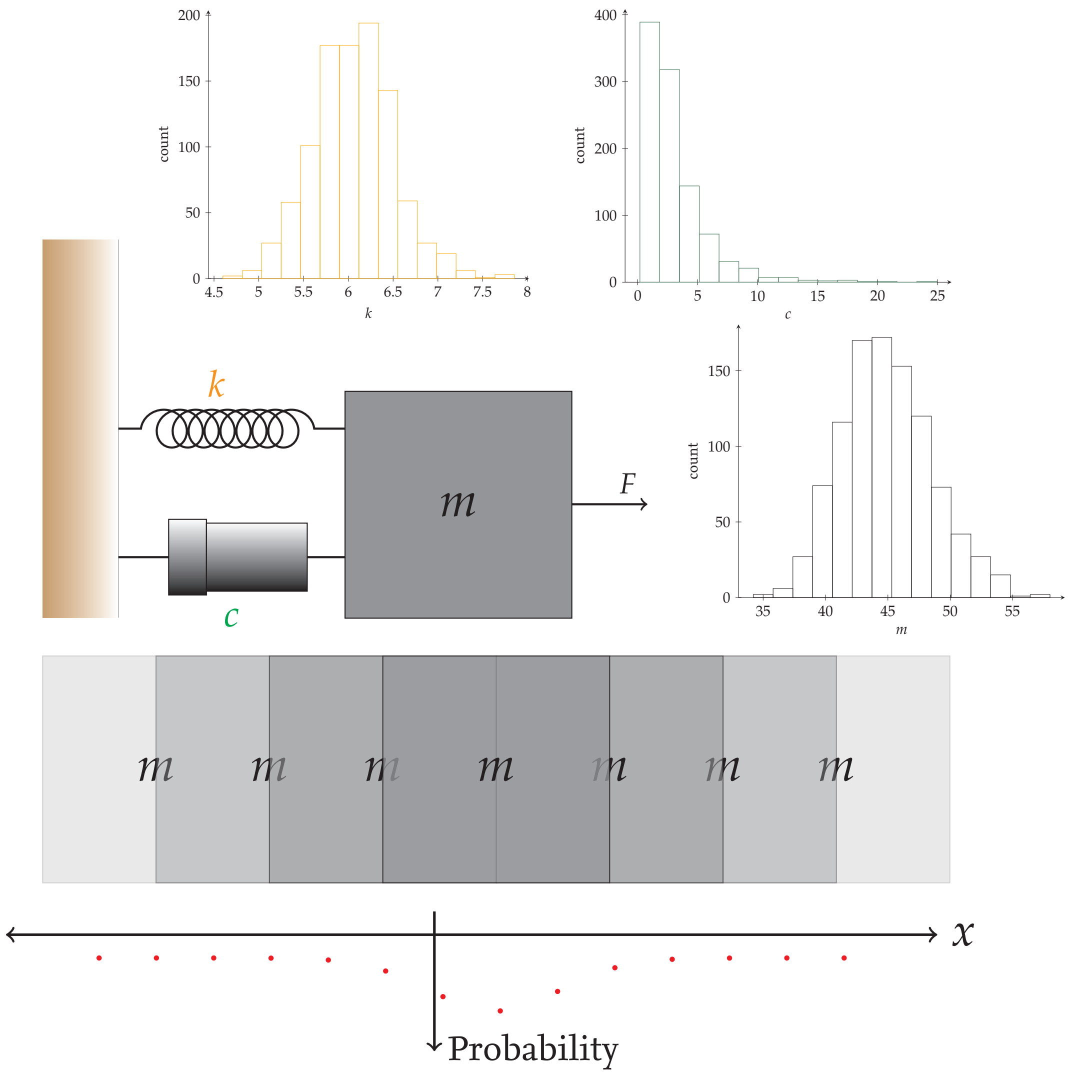


Figure 1.1.: A spring mass damper system with uncertain parameters. Probability distributions of parameters k (orange), c (green) and m (black) and resulting distribution of block position x (red). The position of block at some time $t = t_1$ in gray blocks with opacity proportional to probability.

As seen in Figure 1.1, the resultant position of the block is no longer deterministic but rather probabilistic. The darker shades of the block represent higher probability and the lighter shades represent lower probability and the probability of the position of the block is plotted as shown in the graph (in red) below the blocks [1].

The above illustration is an example of forward propagation of uncertainties, which is

[1]Displacement plot not based on actual computations and is exaggerated for illustration.

one main area in UQ. Here, the uncertainties in the model input are propagated through the model to compute some Quantity of Interest (QoI), such as the mean and/or variance of the position of the block in the above example. A detailed example can be found in section 1.2.4.

Another area of focus in UQ is *model calibration*, also referred to as *parameter identification* or *inverse problems*. In this setting we are interested in inferring the parameters of a model given some observations from experiments. In the above spring-mass-damper illustration, one can estimate the parameters (m), (k) and (c) to certain degree of accuracy given observations of the block's position x.

A third area of focus in UQ is *Sensitivity Analysis* (SA). SA aims to apportion the output uncertainties of a model to uncertainties in the model input [24]. In the above illustration, one can compute the uncertainty induced in the model output by each of the 3 parameters (and their combination) with the help of SA. The benefits of these computations are two-fold. One, we can identify which parameter is responsible for the most model output uncertainty and focus on reducing it. Second, we can choose which parameters can be set of their nominal value, to reduce the overall uncertainty. A detailed example can be found in section 3.

In this work all 3 of the above mentioned problem areas will be covered (with varying degree of detail). However, tasks in UQ are not limited to the above 3 settings but also include *Rare Event Estimation*, *Decision Making* and so on.

1.2. Probability Primer

As seen in the introductory example in the previous section, probability theory plays a central role in UQ. This section is therefore dedicated to reviewing some key concepts and ideas from probability theory.

In the 20th century, modern Probability theory was rigorously established by connecting it to key areas of mathematics such as Measure Theory and Functional Analysis. As of 2021, many texts such as [11] provide a self-contained introduction to modern probability theory.

1.2.1. Sample Space and Probability

In the language of probability, the result of an experiment is called an *outcome* denoted by ω. The collection of all possible outcomes of an experiment is called the *sample space* Ω. An *event* $\mathcal{E}$ is a subset of the sample space and usually represents outcome(s) of interest. Probability theory assigns a number in $[0, 1]$ to all events, which is called the *probability* of that event. Intuitively, it is the likelihood of occurrence of that event. When the sample space is discrete, the probability of an event $\mathcal{E}$, denoted by $\mathbb{P}(\mathcal{E}) := \frac{|\mathcal{E}|}{|\Omega|}$, where $|.|$ denotes the cardinality of the set [11].

Example 1.2.1: Consider rolling a six-sided die once. The sample space can be written as $\Omega = \{1,2,3,4,5,6\}$. The event of rolling a "5" is given by the subset $A = \{5\}$ and the probability $\mathbb{P}(A) = 1/6$ and the event of an odd roll $B = \{1,3,5\}$ has the probability $\mathbb{P}(B) = 3/6$.

Remark: The probabilities computed in the above example must be interpreted using the *strong law of large numbers*. Intuitively, it means that the probabilities will converge to the computed values when we repeat the experiments a large number of times.

1.2.2. Random Variables

Definition 1.2.1 (Random Variables)**.** Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, with a sample space Ω, $\sigma-$algebra on Ω denoted by $\mathcal{F}$, and a probability measure $\mathbb{P}$. A function $X : \Omega \to E \subseteq \mathbb{R}$ is called a random variable, such that for all $x \in E$,

$$\{\omega \mid X(\omega) = x\} \in \mathcal{F}. \tag{1.2}$$

If E is a countable set, then X is called a *discrete* random variable and if E is an uncountable set, then X is called a *continuous* random variable.

Example 1.2.2: Consider a sample space with 6 people $\Omega = \{P_1, P_2, P_3, P_4, P_5, P_6\}$. A random variable $X_1 = \{32, 5, 14, 22, 67, 31\}$ models the age of the group and a second random variable $X_2 = \{0, 1, 0, 1, 1, 0\}$ models if they are football fans.

Random variables help us extract quantities of interest from the elements of the sample space $\omega \in \Omega$, as illustrated in the example above. Typically, we are interested in the values assumed by the random variable, such as age and interest in the above example, rather than modeling the observation space itself [23].

Example 1.2.3: Continuing from Example 1.2.2, the event A of finding a football fan is $A = \{\omega \mid X_2(\omega) = 1\} = \{P_2, P_4, P_5\}$. The probability of finding a fan in the group would therefore be given by $\mathbb{P}(A) = \mathbb{P}(\{\omega \mid X_2(\omega) = 1\}) = \frac{|A|}{|\Omega|} = 3/6$.

Definition 1.2.2 (Probability Mass Function)**.** The PMF of a discrete random variable X assuming values in E is defined as follows:

$$PMF_X := \mathbb{P}(X = x),\ x \in E. \tag{1.3}$$

Definition 1.2.3 (Cumulative Distribution Function)**.** The CDF of a random variable X is defined as follows:

$$F_X(c) := \mathbb{P}(\{X \leq c\}),\ c \in E. \tag{1.4}$$

For a continuous random variable, the set E from Definition 1.2.1 can be uncountably infinite and the PMF becomes the Probability Density Function (PDF). In contrast to the PMF, the PDF is measured in an interval.

Definition 1.2.4 (Probability Density Function)**.** The PDF of a continuous random variable X is defined as follows:

$$\int_a^b f_X(x)\ \mathrm{d}x := \mathbb{P}(\{\omega \mid a \leq X(\omega) \leq b\}). \tag{1.5}$$

Definition 1.2.5 (Expected Value)**.** The expected value of a continuous random variable X is defined as follows:

$$\mathbb{E}[X] := \int_\Omega X(\omega)\ \mathrm{d}\mathbb{P}(\omega) = \int_\mathbb{R} x f_X(x)\ \mathrm{d}x. \tag{1.6}$$

In case of a discrete random variable,

$$\mathbb{E}[X] := \sum_{i=1}^{|X|} x_i PMF_X(X = x_i). \tag{1.7}$$

Intuitively, the expected value a.k.a *weighted* mean is the center of mass of the distribution and is an important measure of central tendency. From the linearity property of the integral operator it follows,

$$\mathbb{E}[\alpha X + Y] = \alpha\mathbb{E}[X] + \mathbb{E}[Y]. \tag{1.8}$$

And for two independent random variables X and Y, the following property holds,

$$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]. \tag{1.9}$$

Definition 1.2.6 (Variance)**.** For a square integrable random variable X, the variance is defined as follows:

$$\mathbb{V}[X] := \mathbb{E}[(X - \mathbb{E}[X])^2]. \tag{1.10}$$

Using linearity of expectation (1.8), variance can also be written as:

$$\begin{aligned}\mathbb{V}[X] &= \mathbb{E}[(X^2 + \mathbb{E}[X]^2 - 2X\mathbb{E}[X])] \\ &= \mathbb{E}[X^2] + \mathbb{E}[\mathbb{E}[X]^2] - 2\mathbb{E}[X] \cdot \mathbb{E}[X] \\ &= \mathbb{E}[X^2] + \mathbb{E}[X]^2 - 2\mathbb{E}[X]^2.\end{aligned}$$

$$\mathbb{V}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2. \tag{1.11}$$

The variance of a random variable tells us about the "spread" of the values around the mean. A small variance corresponds a distribution concentrated around the mean and a large variance indicates that the distribution has a large spread around the mean. Another common metric to describe the spread is the standard deviation, $std[X] := \sqrt{\mathbb{V}[X]}$.

Example 1.2.4: Continuing from Example 1.2.2, the expectation of random variable X_1 $\mathbb{E}[X_1] = \sum_{i=1}^{|\Omega|} x_1^{(i)} PMF_X(x_1^{(i)})$, which is $(32 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 14 \cdot \frac{1}{6} + 22 \cdot \frac{1}{6} + 67 \cdot \frac{1}{6} + 31 \cdot \frac{1}{6}) =$

28.5. The variance [2] can be computed similarly, $\mathbb{V}[X_1] = \sum_{i=1}^{|\Omega|}(x_1^{(i)} - \mathbb{E}[X_1])^2 PMF_X(x_1^{(i)})$ which is 384.25.

It is now useful to consider some common types of probability distributions such as the normal distribution and the uniform distribution. Understanding the properties of probability distributions allow us to model uncertainties that are physically meaningful. For example, the log-normal distribution only assumes positive values, and therefore it is reasonable to employ it to model material properties which are always positive. The choice of distribution clearly requires some domain knowledge.

Normal Distribution

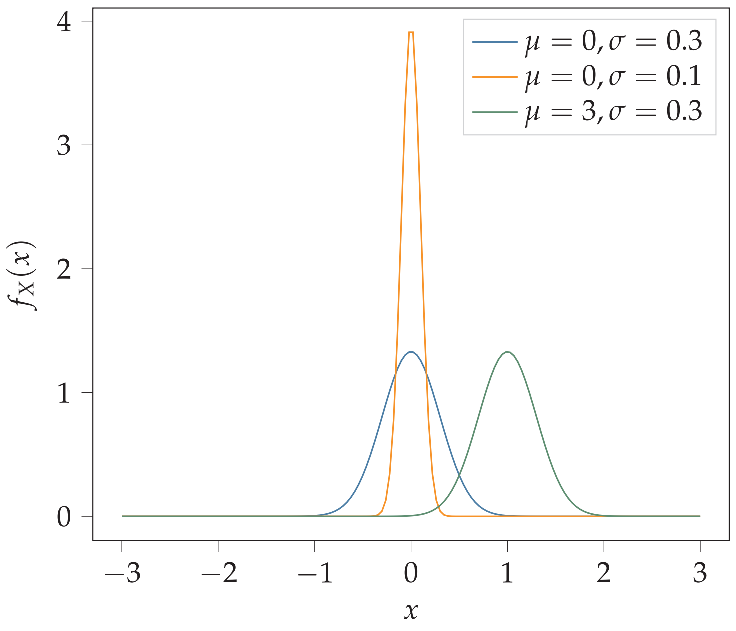


Figure 1.2.: Normal distribution curves with different parameters

A normal distribution, also known as a Gaussian distribution (or informally, a bell curve), is denoted as $X \sim \mathcal{N}(\mu, \sigma^2)$, where μ is the mean of the distribution and σ is the standard deviation. The distribution is completely characterised by these two parameters. The probability distribution function of the Gaussian is

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \tag{1.12}$$

Gaussian distributions with different parameters are illustrated in Figure 1.2. The mean

[2]Technically, this is the biased estimator of the variance, for an unbiased estimator we must use $\mathbb{V}[X_1] = \frac{1}{|\Omega|-1}\sum_{i=1}^{|\Omega|}(x_1^{(i)} - \mathbb{E}[X_1])^2$ which yields a value of 461.1. The difference between the biased and unbiased variance estimator is large in this case, however, the difference reduces when the number of samples is large.

μ governs the center of the curve and the standard deviation σ controls the spread around the mean.

It is typically observed when studying physical characteristics of a population such as weight, height etc. or errors in a physical experiment.

Uniform Distribution

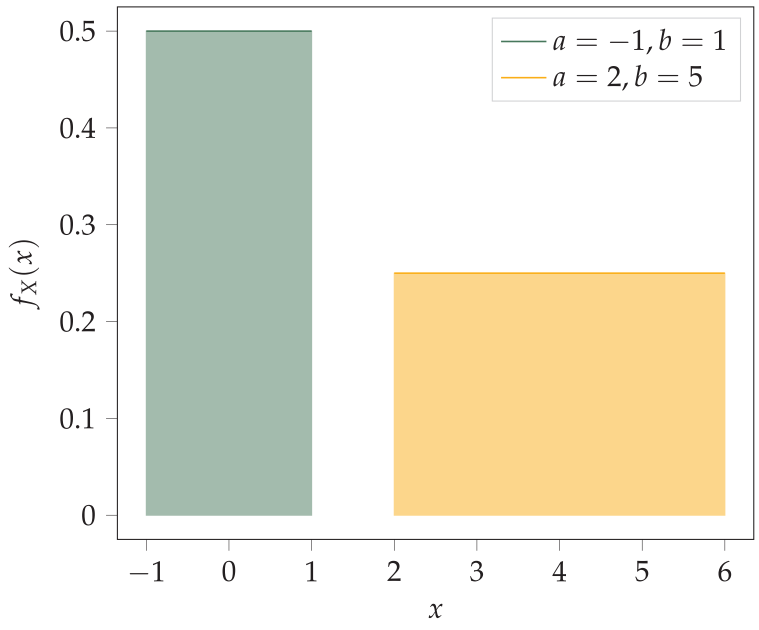


Figure 1.3.: Uniform distribution curves with different parameters

A uniform distribution, denoted as $X \sim \mathcal{U}[a,b]$ only assumes values in the interval $[a,b]$, also known as the support of the distribution and all values in the interval have equal probability. The PDF is given as follows:

$$f_X = \begin{cases} \frac{1}{(b-a)} & x \in [a,b] \\ 0 & \text{else.} \end{cases} \tag{1.13}$$

The mean and variance of the uniform distribution are given as in eq.(1.14) and eq.(1.15) respectively.

$$\mathbb{E}[X] = \frac{b+a}{2}. \tag{1.14}$$

$$\mathbb{V}[X] = \frac{(b-a)^2}{12}. \tag{1.15}$$

Uniform distributions with different parameters are plotted in Figure 1.3.

Multivariate Distributions

We are often interested in analysing several random variables and the relation between them. Therefore it becomes imperative to define *multivariate distributions*, which refers to the probability distribution of several random variables. We shall first consider 2 random variables, X and Y, defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and then later generalise it for n random variables.

Definition 1.2.7 (Covariance). For a square integrable random variables $X : \Omega \to \mathbb{R}$ and $Y : \Omega \to \mathbb{R}$, the covariance is defined as follows:

$$cov[X, Y] := \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]. \tag{1.16}$$

where, the second equality follows from the linearity of the expectation operator (1.8). From the definition of the expectation operator (1.6), we can compute the term $\mathbb{E}[XY]$ as follows:

$$\mathbb{E}[XY] = \int_{\Omega} X(\omega)Y(\omega)\ \mathrm{d}\mathbb{P}(\omega) = \int_{\mathbb{R}}\int_{\mathbb{R}} x\,y\, f_{X,Y}(x, y)\ \mathrm{d}y\ \mathrm{d}x.$$

Another important metric related to covariance is the *correlation coefficient* $\varrho \in [-1, 1]$, which is defined as follows:

$$\varrho[X, Y] = \frac{cov[X, Y]}{std[X]std[Y]}. \tag{1.17}$$

The random variables are termed uncorrelated if $cov[X, Y] = 0$, which implies $\varrho[X, Y] = 0$. $\varrho[X, Y] = 1$ and $\varrho[X, Y] = -1$, are called positive and negative correlation respectively.

A stronger notion of correlation is *independence*. For two independent random variables X and Y, their joint PDFs can be decomposed as $f_{X,Y}(x, y) = f_X(x) f_Y(y)$ and from that it follows, $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$, which implies $\varrho[X, Y] = 0$. The corollary is however not true, i.e. uncorrelated random variables are not necessarily independent (except in the case of Gaussian random variables, which are independent if they are uncorrelated) [23].

Example 1.2.5: Consider two random variables $X \sim \mathcal{U}(-1, 1)$ and $Y = X^2$. We can compute the covariance of X and Y using (1.2.7).

$$cov[X, Y] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] = \mathbb{E}[X^3] - \mathbb{E}[X]\mathbb{E}[X^2].$$

Using the definition of the expectation operator, $\int_{-1}^{1} X^3 - \int_{-1}^{1} X \cdot \int_{-1}^{1} X^2 = 0$ (integration of odd functions with symmetric limits around origin). Clearly, X and Y are dependent but not correlated. (This however changes, if the support of X is changed!)

For a general random vector $\boldsymbol{X} = [X_1, X_2, ..., X_n]^T : \Omega \to \mathbb{R}^n$, where $X_i : \Omega \to \mathbb{R}$, we can define the expectation $\mathbb{E}[\boldsymbol{X}] = [\mathbb{E}[X_1], \mathbb{E}[X_2], ..., \mathbb{E}[X_n]]^T$ and the variance as $\mathbb{V}[\boldsymbol{X}] = [\mathbb{V}[X_1], \mathbb{V}[X_2], ..., \mathbb{V}[X_n]]^T$. The notion of covariance can similarly be extended as $cov[\boldsymbol{X}]_{i,j} = cov[X_i, X_j]$. The covariance matrix is a symmetric matrix in $\mathbb{R}^{n\times n}$, as $cov[X_i, X_j] = cov[X_j, X_i]$ which implies it is non-negative definite [23].

Multivariate Gaussian

If all elements of the random vector $\boldsymbol{X} = [X_1, X_2, ..., X_n]^T$ are normally distributed, then the joint distribution is given as: $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu_X}, \boldsymbol{C_X})$, where $\boldsymbol{\mu_X}$ and $\boldsymbol{C_X}$ refer to the mean and covariance of the multivariate Gaussian respectively. The joint PDF of the multivariate Gaussian is

$$f_X(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^n det(\boldsymbol{C_X})}} exp\Big\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_X})^T \boldsymbol{C_X}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_X})\Big\}. \tag{1.18}$$

For independent Gaussian random variables, the covariance is zero and the covariance matrix $\boldsymbol{C_X}$ is a diagonal matrix and the joint PDF is simply the product of the uni-variate Gaussians.

1.2.3. Random Processes

Random processes, also known as stochastic processes, extend the idea of random variables by including an additional variable, such as time. Common examples of random processes are price of stock, temperature in a region etc.

Definition 1.2.8 (Random Process). A random process can be seen as a collection of infinitely-many random variables $\{X(t) : t \in \mathcal{T} \subset \mathbb{R}\}$, where $X(t)$ is short for $X(t, \omega), \omega \in \Omega$.

$$X : \mathcal{T} \times \Omega \to \mathbb{R}. \tag{1.19}$$

In order to breakdown the idea, we shall once again consider the mechanical oscillator example from the introductory section 1.1 with a slight change (eq. (1.20)). The example considered here is from [3] and will be used for illustrations throughout this work.

$$\begin{aligned} &\ddot{y} + 2\alpha\dot{y} + (\alpha^2+\beta^2)y = 0,\\ &y(0) = \ell,\ \dot{y}(0) = 0. \end{aligned} \tag{1.20}$$

For the purposes of this illustration, the parameters α, β, ℓ will be modeled as random variables, with $\alpha \sim \mathcal{U}(3/8, 5/8)$, $\beta \sim \mathcal{U}(10/4, 15/4)$ and $\ell \sim \mathcal{U}(-5/4, -3/4)$. The solution to the initial value problem (1.20) is given as,

$$y(t; \alpha, \beta, \ell) = \ell e^{-\alpha t}(cos\beta t + \frac{\alpha}{\beta} sin\beta t). \tag{1.21}$$

where, $y(t; \boldsymbol{\xi}(\omega))$ is a random process as defined in (1.19), with $\mathcal{T} = [0, 10]$ and the random vector $\boldsymbol{\xi}$ refers to the collection of random variables α, β, ℓ. We omit ω for simplicity.

In Figure 1.4 we plot the random process y for 3 different realisations of the random vector $\boldsymbol{\xi}$, say $y(t, \boldsymbol{\xi}^{(1)})(blue)$, $y(t, \boldsymbol{\xi}^{(2)})(orange)$ $y(t, \boldsymbol{\xi}^{(3)})(green)$ at $t \in \{0, 1, .., 10\}$. In this

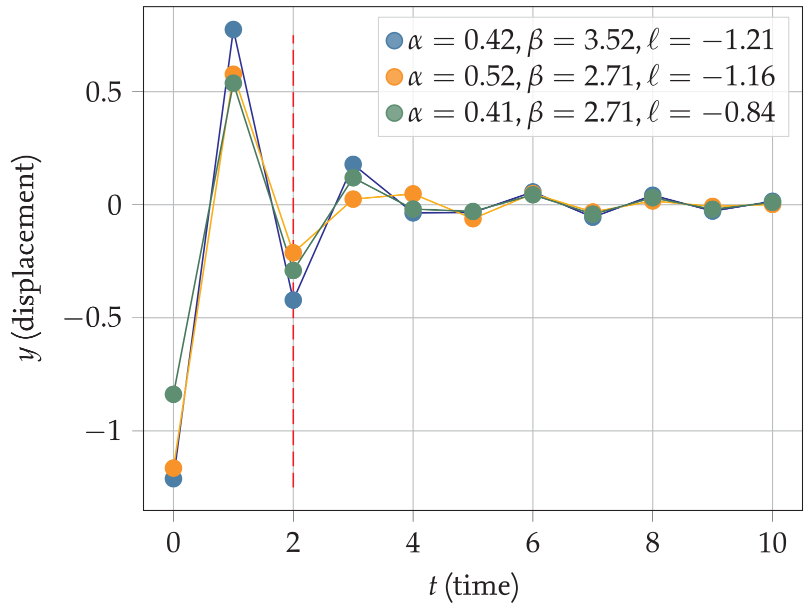


Figure 1.4.: Random Process Example: Mechanical oscillator with 3 realisations.

case, the three curves can be viewed as realisations of the random process, denoted by $y(\cdot, \xi^{(i)})$, where $i \in \{1, 2, 3\}$.

There is however, an alternate perspective of viewing a random process, as given in Definition 1.2.7. In Figure 1.4, consider the process at $t = 2s$ (marked by the red line). We observe three points corresponding to three different realisations of the random vector ξ, $(\xi^{(1)}(blue), \xi^{(2)}(orange), \xi^{(3)}(green))$. Therefore, for a given time t_i, the random process can be viewed as a random variable, namely $y(t_i, \cdot)$ and an infinite collection of such random variables constitutes a random process.

In order to draw a more complete mental picture of a random process, we plot the mechanical oscillator example with 50 realisations in Figure 1.5. The Figure makes the stochastic nature of the process more evident but one can always go back to the above mentioned perspectives as necessary.

We shall now extend the notions of mean and covariance introduced for random variables to random processes.

Definition 1.2.9 (Second-order Random Process)**.** A random process in the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is called a second-order random processes, if the following condition is satisfied [11],

$$\mathbb{E}[|X(t)|^2] < \infty\,, t \in \mathcal{T}. \tag{1.22}$$

Intuitively, this condition requires that the random process be square integrable w.r.t the measure $\mathbb{P}$, alternatively, $X(t) \in L^2(\Omega)$. If the above condition holds, then the mean function $m(t) : \mathcal{T} \to \mathbb{R}$ and the covariance function $\Gamma(t, s) : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ can be defined as follows [11]:

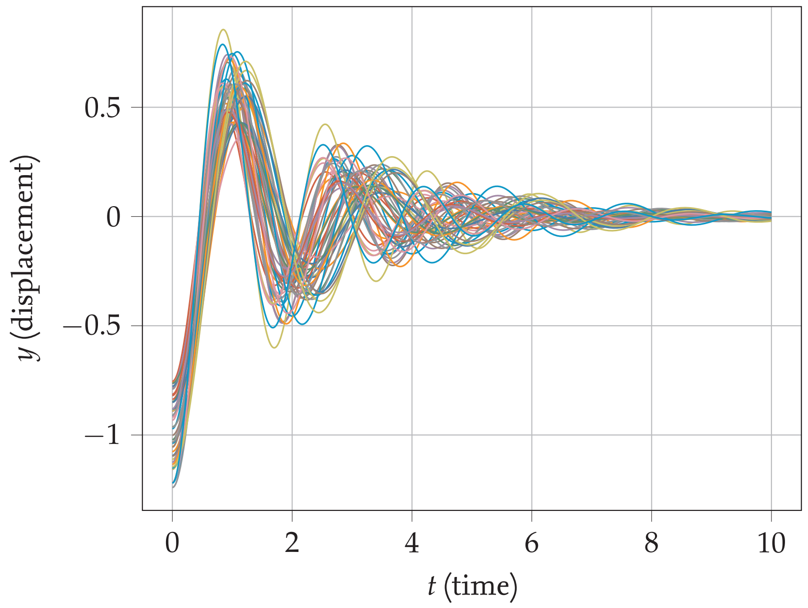


Figure 1.5.: Random Process Example: Mechanical oscillator with 50 realisations.

$$m(t) := \mathbb{E}[X(t)]. \tag{1.23}$$

For a *centered* random process, the mean function is null.

$$\Gamma(t,s) := cov(X(t), X(s)) = \mathbb{E}[X(t)X(s)] - m(t)m(s). \tag{1.24}$$

It is useful to center the random process by subtracting the mean from the realisation to compute the covariance [see for example, 1.7].

$$X_c(t;\xi) := X(t;\xi) - \mathbb{E}[X(t;\xi)]. \tag{1.25}$$

1.2.4. Monte Carlo Methods

In practice, we often encounter PDFs of random variables and random processes which do not have a close form expression and even if the closed form exists, it is often tedious/not possible to compute the mean, covariance and other QoIs analytically. We therefore resort to a class of numerical techniques called the *Monte Carlo (MC) methods*, which essentially use the law of large numbers to compute QoIs such as mean and variance [31].

To demonstrate the use of MC methods, we shall compute the mean function of the random process example from the previous section [3].

The expectation of a random variable X can be approximated using a finite number of samples from the underlying distribution as follows:

[3]It can beshown that the solution (1.21) satisfies the condition (1.22) using the properties (1.8), (1.9) of the expectation operator

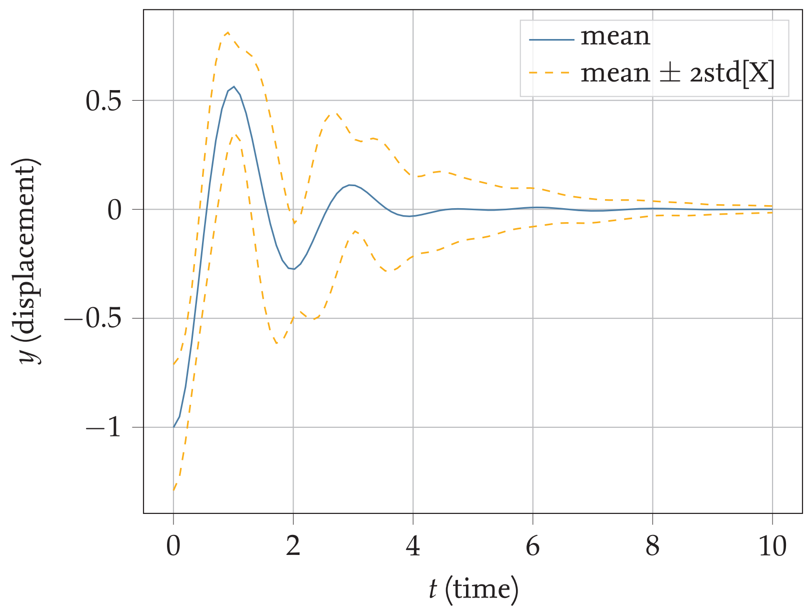


Figure 1.6.: Mean and standard deviation of the mechanical oscillator example

$$\mathbb{E}[X] \approx \tilde{\mathbb{E}}[X] := \frac{1}{N}\sum_{k=1}^{N} x^{(k)},\ x^{(k)} \sim X, \tag{1.26}$$

which follows from (1.6). Here the weighting of variable against the PDF $f_X(x)$ in eq.(1.6) is replaced by picking samples from the distribution X. The approximation $\tilde{\mathbb{E}}[X]$ converges to $\mathbb{E}[X]$ as $N \to \infty$ [31].

To compute the mean function of the random process, we shall use the perspective in definition 1.2.7, which states that a random process can be viewed as a collection of infinite random variables. We consider the process at $t = 2$ as depicted in Figure 1.4 and denote the random variable by $y_{t=2}$. Using the above definition, we can compute the mean of the random process y at $t = 2$s as follows:

$$\mathbb{E}[y_{t=2}] \approx \tilde{\mathbb{E}}[y_{t=2}] = \frac{1}{N}\sum_{k=1}^{N} y(t=2;\xi^{(k)}), \tag{1.27}$$

where $\xi^{(k)} = (\alpha^{(k)}, \beta^{(k)}, \ell^{(k)})$ are drawn from their respective distributions and $y(t = 2; \xi^{(k)})$ refers to the value of the random process (1.21) evaluated for the k^{th} set of random variables. The process is repeated for 101 equidistant points in time in $[0, 10]$ with $N = 20,000$ and is plotted in Figure 1.6 along with the standard deviation which is computed as follows:

$$std[y_{t=2}] \approx \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}\left(y(t=2;\xi^{(k)}) - \tilde{\mathbb{E}}[y_{t=2}]\right)^2}. \tag{1.28}$$

The realisations of the mechanical oscillator example are centered and depicted in Figure 1.7 using the relation (1.25).

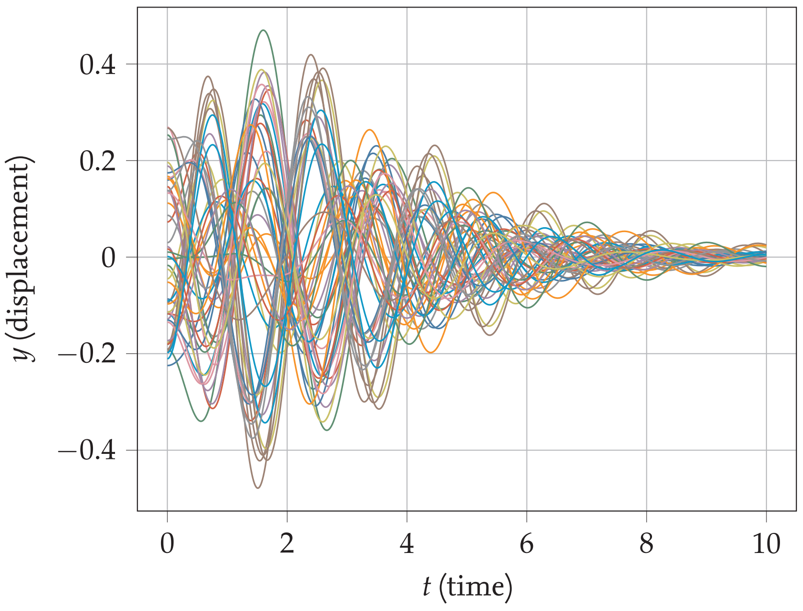


Figure 1.7.: Example of a centered Random Process: Mechanical oscillator with 50 realisations.

Convergence of MC simulations

In the above examples, the QoI(s) are estimated only approximately by MC simulations and contain a sampling error denoted by ϵ_N. Let us for example consider the mean estimator,

$$\mathbb{E}[X] = \frac{1}{N}\sum_{k=1}^{N} x^{(k)} + \epsilon_N,\ x^{(k)} \sim X. \tag{1.29}$$

From the law of large numbers it can be shown that the error ϵ_N asymptotically reaches zero with a convergence rate of $\mathcal{O}(1/\sqrt{N})$. The low convergence rate can be slightly improved by better sampling strategies such as Quasi Monte Carlo sampling (QMC) which has a convergence rate of $\mathcal{O}(\frac{logN^n}{N})$, where n refers to the dimension of the input random variable [19].

Though MC simulations have a low convergence rate, they allow us to circumvent the curse of dimensionality [19], which refers to an exponential increase in the number of input points needed to sufficiently sample the input space. For example, when using uniformly distributed points to sample the input space, the number of points needed are N^d. In the mechanical oscillator example, if we use 100 points per dimension of the input space of dimension $d = 3$, we will need 10^6 model evaluations to estimate the mean.

2. Surrogate Modeling

In the previous section we noted that MC methods though robust and easy to implement, require a large number of model evaluations to compute QoIs. Since a model evaluation only provides local information on the solution manifold, a large number of evaluations become necessary to determine its global statistical properties [19]. In the mechanical oscillator example, the model was computationally cheap to evaluate and we computed the mean and standard deviation using $20,000$ model evaluations. However in practice, models can be very expensive to evaluate and can take anywhere from a few minutes to several hours for a single evaluation and this becomes a bottleneck for UQ studies.

In order to alleviate the computational expense, we resort to approximations of the complex models called *surrogate models* which are computationally cheaper to evaluate. In this work we restrict our focus to the following kind of models,

$$\mathcal{M}(t, \boldsymbol{\xi}) : \mathcal{T} \times \Gamma \to \mathbb{R}, \ \mathcal{T} \subseteq \mathbb{R}, \tag{2.1}$$

where $\mathcal{M}(t, \boldsymbol{\xi})$ is a random process in the probability space $(\Gamma, \mathcal{B}(\Gamma), f_{\boldsymbol{\xi}}(\boldsymbol{\xi})\ \mathrm{d}\boldsymbol{\xi})$ with $\Gamma \subseteq \mathbb{R}^{N_p}$ and $\mathcal{B}(\Gamma)$ is the Borel sigma-algebra on Γ. The probability space $(\Gamma, \mathcal{B}(\Gamma), f_{\boldsymbol{\xi}}(\boldsymbol{\xi})\ \mathrm{d}\boldsymbol{\xi})$ can be mapped to the previous probability space $(\Omega, \mathcal{F}, \mathbb{P})$ using the random vector $\boldsymbol{\xi} = (\xi_1(\omega), \dots, \xi_{N_p}(\omega)) \in \Gamma$, where N_p denotes the number of random variables (For details refer [p.12 31]). Here, the deterministic variable t refers to a temporal coordinate with $\mathcal{T} = [0, T]$. Our goal is to approximate the stochastic solution manifold $Y = \mathcal{M}(t, \boldsymbol{\xi})$ using a surrogate $\tilde{Y} = \tilde{\mathcal{M}}(t, \boldsymbol{\xi})$.

There are several types of surrogate models discussed in literature for UQ, such as *Neural Networks* [33], *Gaussian Process Regression* a.k.a *Kriging* [5], *Polynomial Chaos Expansion* (PCE) [19, 37], *Karhunen-Loève Expansion* (KLE) [19, 2, 6] and so on. In this work, we shall limit our focus to the latter two methods, namely PCE and KLE with PCE approximated modes (KLE+PCE).

The two methods, PCE and KLE+PCE, belong to a general class of methods called *spectral methods*, which approximate the stochastic solution manifold using a basis [19],

- Generalised Polynomial Chaos Expansion:

$$\mathcal{M}(t, \boldsymbol{\xi}) \approx \tilde{\mathcal{M}}^{PC}(t, \boldsymbol{\xi}) = \sum_{k=1}^{N_{PC}} c_k(t) \Psi_k(\boldsymbol{\xi}). \tag{2.2}$$

  where $\{\Psi_k(\boldsymbol{\xi})\}_{k=1}^{N_{PC}}$ represents a multivariate orthonormal polynomial basis and $\{c_k\}_{k=1}^{N_{PC}}$ represent the coefficients.

- Karhunen-Loève Expansion with PCE approximated modes:

$$\mathcal{M}(t,\boldsymbol{\xi}) \approx \tilde{\mathcal{M}}^{KL+PC}(t,\boldsymbol{\xi}) = \sum_{i=1}^{N_{kl}} \Phi_i(\boldsymbol{\xi}) e_i(t). \tag{2.3}$$

where $\{e_i(t)\}_{i=1}^{N_{kl}}$ represents eigenfunctions of the covariance operator of $\mathcal{M}(t,\boldsymbol{\xi})$ and $\Phi_i(\boldsymbol{\xi})$ represents the expansion coefficients, which is called the KLE expansion of the model. In this work, the projections $\Phi_i(\boldsymbol{\xi})$ are approximated using PCE and the model can therefore be written as:

$$\tilde{\mathcal{M}}^{KL+PC}(t,\boldsymbol{\xi}) = \sum_{i=1}^{N_{kl}} \sum_{k=1}^{N_{PC}} f_{\Phi_i}^{(k)} \mathcal{F}_{\Phi_i}^{(k)}(\boldsymbol{\xi}) e_i(t). \tag{2.4}$$

For most real world problems only a discretised version of the random process (2.1) is available. Therefore, it is important to consider a discretised response of the model which we shall denote by $\boldsymbol{Y} = (Y_{t_1}, \ldots, Y_{t_m})$ at time $t_1, \ldots, t_m$. The spectral methods introduced above build surrogates to emulate the model response which we denote by $\tilde{\boldsymbol{Y}} = (\tilde{Y}_{t_1}, \ldots, \tilde{Y}_{t_m})$. The idea is illustrated in Figure 2.1. The gPCE surrogate approximates the model response by building *local surrogates* (a.k.a time-frozen surrogates [20]) at each time point t_s in an orthogonal polynomial basis and the KLE+PCE surrogate first finds a suitable basis to represent the process and builds a *global surrogate*. In this chapter we shall introduce the two methods and illustrate their use with the mechanical oscillator example from the previous section.

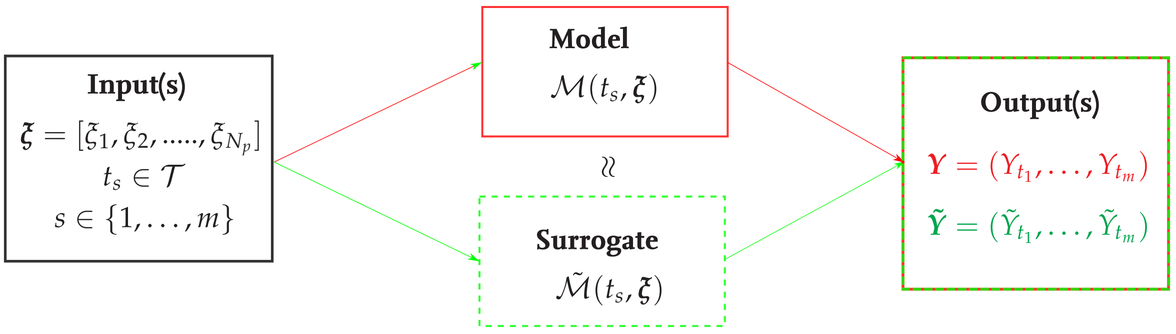

Input(s)
$\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_{N_p}]$
$t_s \in \mathcal{T}$
$s \in \{1, \ldots, m\}$
Model
$\mathcal{M}(t_s, \boldsymbol{\xi})$
$\wr\wr$
Surrogate
$\tilde{\mathcal{M}}(t_s, \boldsymbol{\xi})$
Output(s)
$\boldsymbol{Y} = (Y_{t_1}, \ldots, Y_{t_m})$
$\tilde{\boldsymbol{Y}} = (\tilde{Y}_{t_1}, \ldots, \tilde{Y}_{t_m})$

Figure 2.1.: Surrogate modeling using discretised model response.

2.1. Polynomial Chaos Expansion (PCE)

Before we introduce time-frozen surrogates for time-dependent models, we begin by considering a simple time-independent model $\mathcal{M}(\boldsymbol{\xi})$. (As we will see later, time-frozen surrogates build approximations of $\mathcal{M}(t_i, \boldsymbol{\xi})$ by fixing t to some $t_i \in \mathcal{T}$). For simplicity, we assume it has an independent random input vector $\boldsymbol{\xi}$ with a joint probability distribution $f_{\boldsymbol{\xi}}(\boldsymbol{\xi})$, with support $\mathcal{D}_{\boldsymbol{\xi}}$ defined over a probability space $(\Gamma, \mathcal{B}(\Gamma), f_{\boldsymbol{\xi}}(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi})$ and the scalar output denoted by Y. In order to build a PCE surrogate we require that the model response Y be square integrable. Mathematically [21]:

$$\mathbb{E}[Y^2] = \int_{\mathcal{D}_{\boldsymbol{\xi}}} \mathcal{M}^2(\boldsymbol{\xi})\, f_{\boldsymbol{\xi}}(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} < \infty. \tag{2.5}$$

We now approximate the model $\mathcal{M}(\boldsymbol{\xi})$ in an orthogonal polynomial basis $\{\Psi_k(\boldsymbol{\xi})\}_{k=1}^{N_{PC}}$ as follows:

$$\mathcal{M}(\boldsymbol{\xi}) \approx \tilde{\mathcal{M}}^{PC}(\boldsymbol{\xi}) = \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}), \tag{2.6}$$

where $\{c_k\}_{k=1}^{N_{PC}}$ represent the coefficients in the basis. The following sections summarise the presentation in [21, 29, 27].

2.1.1. Univariate Orthonormal Polynomial Basis

Before we introduce orthogonal polynomials, it is useful to revisit the notion of *orthogonality*. We know from linear algebra that two vectors $\vec{a}$ and $\vec{b}$ are orthogonal if $\langle \vec{a}, \vec{b} \rangle = 0$. In functional analysis, this idea is extended to functions, where two functions, for example, $u, v \in L^2(\mathcal{X}, \mu)$ are said to be orthogonal w.r.t measure μ if,

$$\langle u, v \rangle := \int_{\mathcal{X}} u(x)v(x)d\mu = 0. \tag{2.7}$$

Example 2.1.1.1 Consider a random variable $X \sim \mathcal{N}(0,1)$ with support $\mathcal{D}_X = \mathbb{R}$ and PDF $f_X(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ (refer 1.12). Two functions $\phi_1(x), \phi_2(x) : \mathcal{D}_X \to \mathbb{R}$, for example, $\phi_1(x) := 1$ and $\phi_2(x) := x^2 - 1$ are orthogonal w.r.t the probability measure $\mathbb{P}(\mathrm{d}x) = f_X(x)\,\mathrm{d}x$,

$$\int_{\mathcal{D}_X} \phi_1(x)\phi_2(x)\mathbb{P}(\mathrm{d}x) = \int_{-\infty}^{\infty} 1 \cdot (x^2-1) f_X(x)\,\mathrm{d}x = \int_{-\infty}^{\infty} 1 \cdot (x^2-1) \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\,\mathrm{d}x = 0.$$

In the following subsection we introduce a class of polynomials that are orthogonal w.r.t uniformly distributed random variables called the Legendre Polynomials. For a similar representation of Hermite Polynomials that are orthogonal w.r.t normally distributed random variables (as in the above example), refer [27].

Legendre Polynomials

A family of orthogonal polynomials can be obtained by applying the Gram-Schmidt orthogonalisation to the canonical set of monomials $\{1, x, x^2, \ldots\}$ with the appropriate *weighting function* $f_X(x)$ above.

For example, if $\xi_i \sim \mathcal{U}(-1, 1)$, we apply orthogonalisation as in the above example w.r.t the PDF $f_{\xi_i}(\xi) = \frac{1}{2}$ (obtained using equation (1.13)). The corresponding set of polynomials are called *Legendre Polynomials* and are denoted by $P_n(\xi)$, where $n \in \mathbb{N}_0$ is the degree of the polynomial and $\mathbb{N}_0$ denotes the set of natural numbers (including 0). Refer example in Appendix C.

In general, these polynomials can be normalised using following relations such that $< P_n, P_n >= 1$,

$$\|P_n\|^2 = \langle P_n, P_n \rangle = \int_{-1}^{1} P_n^2(\xi) \cdot \frac{1}{2} \, \mathrm{d}\xi = \frac{1}{2n+1}. \tag{2.8}$$

$$\tilde{P}_n(\xi) = \sqrt{2n+1} P_n(\xi). \tag{2.9}$$

where $\tilde{P}_n(\xi_i)$ denotes a normalised Legendre polynomial of degree n. The first 6 Legendre polynomials are listed below and further polynomials can be generated using the recurrence relation (2.11).

n	$P_n(\xi)$	$\|P_n\|^2$	$\tilde{P}_n(\xi)$
0	1	1	1
1	ξ	$1/3$	$\sqrt{3}P_1$
2	$\frac{1}{2}(3\xi^2 - 1)$	$1/5$	$\sqrt{5}P_2$
3	$\frac{1}{2}(5\xi^3 - 3\xi)$	$1/7$	$\sqrt{7}P_3$
4	$\frac{1}{8}(35\xi^4 - 30\xi^2 + 3)$	$1/9$	$\sqrt{9}P_4$
5	$\frac{1}{8}(63\xi^5 - 70\xi^3 + 15\xi)$	$1/11$	$\sqrt{11}P_5$

(2.10)

$$nP_n(\xi) = (2n-1)\xi P_{n-1}(\xi) - (n-1)P_{n-2}(\xi). \tag{2.11}$$

2.1.2. Multivariate Orthonormal Polynomial Basis

Most distributions we encounter in reality have multiple random variables (usually belonging to different classes of random variables) and thus constructing a multivariate basis becomes necessary. We assume that the random variables are modeled using classical probability distributions such as Gaussian, uniform etc. and an orthogonal family of polynomials exists for each of the random variable that satisfies equation (2.7). A list of classical distributions and their corresponding family of orthogonal polynomials can be found in the Askey-Scheme in [38]. A multivariate polynomial basis created using such polynomials is called generalised Polynomial Chaos Expansion (gPCE).

A multivariate basis can be created by computing the tensor product of the univariate polynomials [29]. Let the polynomial degree corresponding to a random variable be denoted by an ordered list $\boldsymbol{\alpha} = \left(\alpha_1, \dots, \alpha_{N_p}\right)$ where, $\alpha_i \in \mathbb{N}_0$. A multivariate polynomial basis $\Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$ can be given as,

$$\Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}) := \prod_{i=1}^{N_p} \psi_{\alpha_i}^{(i)}(\xi_i), \tag{2.12}$$

where $\psi_{\alpha_i}^{(i)}$ are orthonormal polynomials as defined in equation (2.7). The polynomial basis defined above is orthogonal by construction,

$$\mathbb{E}\left[\Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})\Psi_{\boldsymbol{\beta}}(\boldsymbol{\xi})\right] = \int_{\mathcal{D}_{\xi}} \Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})\Psi_{\boldsymbol{\beta}}(\boldsymbol{\xi}) f_{\xi}(\boldsymbol{\xi}) d\boldsymbol{\xi} = \delta_{\boldsymbol{\alpha\beta}} \quad \forall \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{N}_0^{N_p}. \tag{2.13}$$

In the above relation, $\delta_{\boldsymbol{\alpha\beta}}$ represents the Kronecker delta function, where $\delta_{\boldsymbol{\alpha\beta}} = 1$, if $\boldsymbol{\alpha} = \boldsymbol{\beta}$ and 0 otherwise. We extend the notion of polynomial degree in univariate polynomials to the multivariate case using the idea of *total degree* denoted by $|\boldsymbol{\alpha}|$ which simply sums the degree of each of the univariate polynomials as follows:

$$|\boldsymbol{\alpha}| := \sum_{i=1}^{N_p} \alpha_i. \tag{2.14}$$

Using the multivariate orthonormal polynomials $\Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$, we can form a basis in $L^2(\Gamma, f_{\xi}(\boldsymbol{\xi})\, \mathrm{d}\boldsymbol{\xi})$ which can approximate our model $\mathcal{M}(\boldsymbol{\xi})$,

$$\mathcal{M}(\boldsymbol{\xi}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^{N_p}} c_{\boldsymbol{\alpha}} \Psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}). \tag{2.15}$$

The expansion above contains (countably) infinite number of terms but in reality we must use a finite approximation for our model. Therefore, we use a truncation scheme to limit the number of terms considered in the basis. In the standard truncation scheme, we consider all multivariate polynomials with a total degree less than or equal to p and the corresponding set of polynomial degrees is given as follows:

$$\mathcal{A}^{N_p,p} := \left\{ \boldsymbol{\alpha} \in \mathbb{N}_0^{N_p} \mid |\boldsymbol{\alpha}| \leq p \right\}. \tag{2.16}$$

The elements in the above set are written in lexicographic order. The number of terms in the truncated series N_{PC} is given as [21],

$$N_{PC} := \operatorname{card} \mathcal{A}^{N_p,p} = \frac{(N_p + p)!}{N_p! p!}. \tag{2.17}$$

Finally, we obtain the finite gPCE as introduced in (2.2),

$$\mathcal{M}(\boldsymbol{\xi}) \approx \sum_{k=1}^{N_{PC}} c_k \Psi_{\boldsymbol{\alpha}^k}(\boldsymbol{\xi}) = \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}). \tag{2.18}$$

where $\boldsymbol{\alpha}^k$ denotes the k^{th} element of set $\mathcal{A}^{N_p,p}$ and $\Psi_k(\boldsymbol{\xi})$ is short for $\Psi_{\boldsymbol{\alpha}^k}(\boldsymbol{\xi})$.

2.1.3. Isoprobabilistic transform

Within the regime of gPCE expansions with independent random variables modeled using classical distributions, we encounter non-standard distributions. For example, in the Legendre polynomials described above, the orthonormal polynomials were derived assuming a standard uniform distribution $\xi_i \sim \mathcal{U}(-1,1)$. But the family of orthogonal polynomials would change if the bounds of the distribution are altered and in reality we often encounter such non-standard distributions. In this case, one can proceed in one of two ways [21], either

- Construct an orthonormal polynomial basis for each random variable for the non-standard distribution(s)

  or

- Transform the non-standard distribution(s) to standard distribution(s) using an *isoprobabilistic transform*

In this work we only consider independent random variables modeled using classical distributions and therefore for simplicity, we construct the orthonormal basis once and use an isoprobabilistic transform $\mathcal{I}$ for non-standard distributions.

Example 2.1.2 Consider a random vector $\boldsymbol{\xi} = \{\xi_1, \xi_2\}^T$, where $\xi_1 \sim \mathcal{U}(a,b)$ and $\xi_2 \sim \mathcal{N}(\mu,\sigma)$. In order to convert $\boldsymbol{\xi}$ to $\mathbf{Z} = \{Z_1, Z_2\}^T$, where $Z_1 \sim \mathcal{U}(-1,1)$ and $Z_2 \sim \mathcal{N}(0,1)$, we use an isoprobabilistic transform $\mathcal{I}(\boldsymbol{\xi}) = \mathbf{Z}$,

$$Z_1 = \frac{\xi_1 - \frac{b+a}{2}}{\frac{b-a}{2}}. \tag{2.19}$$

$$Z_2 = \frac{\xi_2 - \mu}{\sigma}. \tag{2.20}$$

The finite gPCE approximation (2.18) can now be re-written as,

$$\mathcal{M}(\boldsymbol{\xi}) \approx \sum_{k=1}^{N_{PC}} c_k \Psi_k(\mathcal{I}(\boldsymbol{\xi})). \tag{2.21}$$

For an example of a finite gPCE expansion with isoprobabilistic transform refer [29, p.9] or [27, p. 32].

NOTE: Going forward it will be assumed that an isoprobabilistic transform has been applied for non-standard random variables and will not be explicitly repeated.

2.1.4. Computing coefficients

The methods to determine the coefficients of a surrogate model, such as c_α in equation (2.21), can be classified into one of two ways,

- *Intrusive methods*, such as the *Stochastic Galerkin Method* [19, 37], alter the deterministic model solver to include the input random variables and thereby "intrude" into the solver
- *Non-intrusive methods* determine the coefficients by repeated model evaluations using the deterministic solver. The number of model evaluations in this case are however smaller than the number of evaluations necessary for the computation of the QoI(s) using MC simulations.

In this work the PCE coefficients are computed using *Least-square minimisation*, a non-intrusive approach. For a quick overview of other methods refer [29, section 3.4]. In literature, computation of surrogate coefficients is also referred to as *training*.

Least-square minimisation

In the finite gPCE expansion (2.18), we truncated the gPCE (2.15) and thereby incurred an error ϵ in our approximation. We can therefore write our finite gPCE expansion as follows:

$$Y = \mathcal{M}(\boldsymbol{\xi}) = \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}) + \epsilon. \tag{2.22}$$

The least-square minimisation method aims to compute the PCE coefficients c_k in (2.18) by minimizing the error ϵ in the least squared sense,

$$\mathbb{E}[\epsilon^2] := \mathbb{E}\left[\left(Y - \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi})\right)^2\right]. \tag{2.23}$$

The minimisation problem can be summarised as follows:

$$\boldsymbol{c} = \underset{\boldsymbol{c}\in\mathbb{R}^{N_{PC}}}{\arg\min}\, \mathbb{E}\left[\left(\mathcal{M}(\boldsymbol{\xi}) - \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi})\right)^2\right]. \tag{2.24}$$

The expectation in the above problem is estimated using N Monte Carlo simulations as in equation (1.26), which implies N model evaluations are necessary. Typically $N \approx (N_p - 1)N_{PC}$ [p.970 28]. Minimising the estimated residual yields an estimate of the coefficients $\hat{\boldsymbol{c}}$,

$$\hat{\boldsymbol{c}} = \underset{\boldsymbol{c}\in\mathbb{R}^{N_{PC}}}{\arg\min}\, \frac{1}{N} \sum_{i=1}^{N} \left(\mathcal{M}(\boldsymbol{\xi}^{(i)}) - \sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}^{(i)})\right)^2. \tag{2.25}$$

The above minimisation problem can be solved using *normal equations*. We begin by writing the above in vector notation (here the constant $\frac{1}{N}$ can be ignored since we only seek the minimizer of the argument and not the minimum itself),

$$\hat{\boldsymbol{c}} = \underset{\boldsymbol{c}\in\mathbb{R}^{N_{PC}}}{\arg\min}\, (\mathcal{Y} - \mathbf{A}\boldsymbol{c})^2. \tag{2.26}$$

where the vector $\mathcal{Y} \in \mathbb{R}^N$ contains the model evaluations,

$$\mathcal{Y} = \left\{ y^{(1)} = \mathcal{M}\left(\boldsymbol{\xi}^{(1)}\right), \dots, y^{(N)} = \mathcal{M}\left(\boldsymbol{\xi}^{(N)}\right) \right\}^\top, \tag{2.27}$$

and the matrix $\Xi \in \mathbb{R}^{N \times N_p}$ contains the MC samples,

$$\Xi := \left\{ \boldsymbol{\xi}^{(1)}, \dots, \boldsymbol{\xi}^{(N)} \right\}^\top. \tag{2.28}$$

The matrix $\mathbf{A} \in \mathbb{R}^{N \times N_{PC}}$ is the *Vandermonde matrix* whose elements a_{ij} are computed as follows:

$$a_{ij} := \Psi_j(\boldsymbol{x}^{(i)}), \quad i = \{1, \dots, N\}, \quad j = \{1, \dots, N_{PC}\}. \tag{2.29}$$

The solution to (2.26) is given by the normal equations as follows:

$$\hat{c} = \left(\mathbf{A}^\top \mathbf{A}\right)^{-1} \mathbf{A}^\top \mathcal{Y}. \tag{2.30}$$

where $\left(\mathbf{A}^\top \mathbf{A}\right)^{-1} \mathbf{A}^\top$ is called the *Moore–Penrose inverse*. Using the newly computed surrogate coefficients, we can approximate the response of the model for any random variable.

$$\tilde{\mathcal{M}}^{\mathrm{PC}}(\boldsymbol{\xi}) = \sum_{k=1}^{N_{PC}} \hat{c}_k \Psi_k(\boldsymbol{\xi}). \tag{2.31}$$

2.1.5. Leave-one-out cross-validation

In order to validate the surrogate, we must estimate the error ϵ in (2.22). In this work we estimate the error using the *Leave-one-out* scheme and denote it by ϵ_{LOO} [21]. In this scheme we build a PCE surrogate $\tilde{\mathcal{M}}^{PC\backslash i}$ by excluding a sample $\boldsymbol{\xi}^{(i)}$ from (2.28). Once the PCE is built we evaluate the response for the excluded sample $\boldsymbol{\xi}^{(i)}$ and compute the difference with the true model response. We repeat this procedure for all samples and divide the sum by the model variance to obtain a relative error estimate,

$$\epsilon_{LOO} := \frac{\sum_{i=1}^N \left(\mathcal{M}\left(\boldsymbol{\xi}^{(i)}\right) - \tilde{\mathcal{M}}^{PC\backslash i}\left(\boldsymbol{\xi}^{(i)}\right) \right)^2}{\sum_{i=1}^N \left(\mathcal{M}\left(\boldsymbol{\xi}^{(i)}\right) - \hat{\mu}_Y \right)^2}, \tag{2.32}$$

where $\hat{\mu}_Y$ is mean response of the model given by,

$$\hat{\mu}_Y := \frac{1}{N} \sum_{i=1}^N \mathcal{M}\left(\boldsymbol{\xi}^{(i)}\right). \tag{2.33}$$

An efficient method to compute ϵ_{LOO} is given in [7] and is summarised in [21] as follows:

$$\epsilon_{LOO} = \frac{\sum_{i=1}^N \left(\dfrac{\mathcal{M}\left(\boldsymbol{\xi}^{(i)}\right) - \tilde{\mathcal{M}}^{PC}\left(\boldsymbol{\xi}^{(i)}\right)}{1 - h_i} \right)^2}{\sum_{i=1}^N \left(\mathcal{M}\left(\boldsymbol{\xi}^{(i)}\right) - \hat{\mu}_Y \right)^2}, \tag{2.34}$$

where h_i is the i^{th} component of the vector $\boldsymbol{h}$ which can be obtained as follows,

$$\boldsymbol{h} = \operatorname{diag}\left(\mathbf{A}\left(\mathbf{A}^\top\mathbf{A}\right)^{-1}\mathbf{A}^\top\right). \tag{2.35}$$

2.1.6. Time-frozen surrogates

In the previous sections we computed a surrogate for a time-independent model $\mathcal{M}(\boldsymbol{\xi})$. In order to construct a surrogate for a time-dependent process $\mathcal{M}(t, \boldsymbol{\xi})$, we can construct local surrogates for each point t_s for $s \in \{1, ..., m\}$, which are termed *time-frozen surrogates* [20]. For a value $t = t_s$, we obtain a model $\mathcal{M}(t_s, \boldsymbol{\xi})$ which can be approximated as in (2.18). We recollect the equation here for convenience,

$$\mathcal{M}(t_s, \boldsymbol{\xi}) \approx \tilde{\mathcal{M}}^{PC}(t_s, \boldsymbol{\xi}) = \sum_{k=1}^{N_{PC,s}} c_{k,s}\Psi_{k,s}(\boldsymbol{\xi}), \tag{2.36}$$

where each local surrogate can have a different number of basis terms $N_{PC,s}$ based on the complexity of the stochastic solution manifold $\mathcal{M}(t_s, \boldsymbol{\xi})$.

To illustrate, we compute $m = 51$ time-frozen surrogates in $\mathcal{T} = [0, 10]$ for the mechanical oscillator example from chapter 1. In Figure 2.2, we indicate 6 sample time points $t_s \in \{0, 2, 4, 6, 8, 10\} \subset \mathcal{T}$ at which the surrogates are constructed. Each surrogate has a total polynomial degree $p = 5$ and the coefficients are computed using $20,000$ Monte Carlo samples. Since the random variables α, β, ℓ are modelled using uniform random variables, Legendre polynomials are used to construct the polynomial basis. Using the relation (2.17), we can compute the total number of terms in the basis $N_{PC} = \frac{(N_p+p)!}{N_p!p!} = \frac{(3+5)!}{3!5!} = 56$.

In order to examine the accuracy of the approximations we compute the Leave-One-Out error (ϵ_{LOO}) for each of the 51 surrogates and plot it in Figure 2.3. We know that increasing the total polynomial degree p increases the approximation capacity of the surrogate. Therefore we compute the time-frozen surrogates with an increased polynomial degree of $p = 7$ which leads to a polynomial basis with $N_{PC} = \frac{(N_p+p)!}{N_p!p!} = \frac{(3+7)!}{3!7!} = 120$ and plot the Leave-One-Out error in same Figure to highlight the effect.

Drawbacks

The local surrogates approximate the solution manifold $\mathcal{M}(t_s, \boldsymbol{\xi})$ at time $t = t_s$ using a polynomials basis. This method however has some key drawbacks,

- The approximation quality of time-frozen PCEs deteriorates with time due to the increasing complexity of the stochastic process [summarised in 20]. See Figure 2.3.
- Creating local surrogates becomes quickly intractable when a large number of local surrogates are required.
- Since a local polynomial basis is created, the method does not exploit the underlying structure of the time-dependent model.

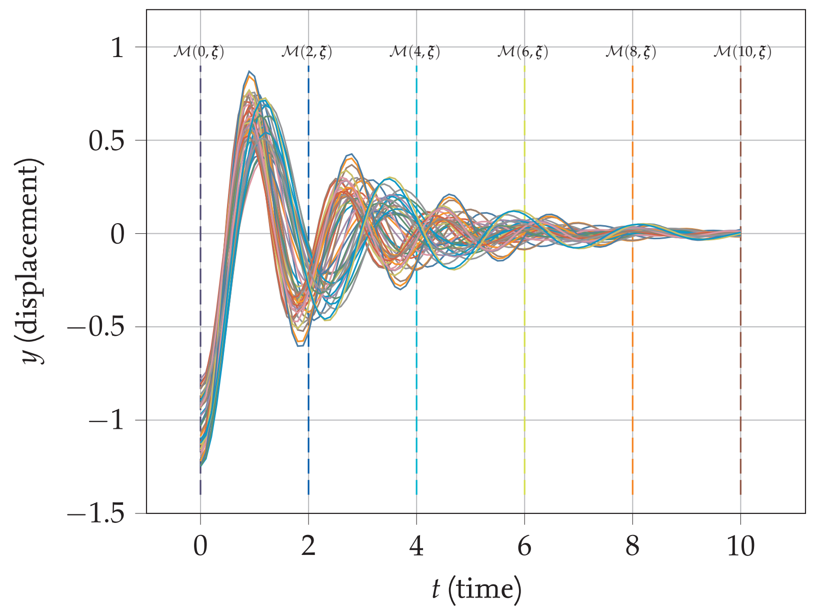


Figure 2.2.: Time frozen PCE surrogates for the mechanical oscillator example at $t_s \in \{0, 2, 4, 6, 8, 10\}$ each with $p = 5$ and $N_{PC} = 56$ trained using $20,000$ MC samples.

2.2. Karnhunen-Loève Expansion with PCE approximated modes

In order to address the drawbacks in the time-frozen PCE approach, a global surrogate model using Karnhunen-Loève Expansion with PCE approximated modes (KLE+PCE) is proposed in [6]. The main idea of the method is to exploit the underlying structure of the random process. The surrogate model uses the eigenfunctions of the covariance operator to represent the random process $\mathcal{M}(t, \xi)$. Although the basis is sometimes difficult to interpret, it is mathematically more efficient since usually only a small number of eigenfunctions are necessary to obtain a good approximation of the random process. One may be more familiar with discrete version of KLE, often termed *Principal Component Analysis* (PCA) or *Proper Orthogonal Decomposition* (POD).

The rest of this section is organised as follows: we briefly discuss the mathematical framework of KLE in section 2.1 and then apply the decomposition to the model in the finite dimensional case and provide an overview for practical implementation in section 2.2. In order to assess the quality of the approximation, we briefly discuss error estimates in section 2.3. Finally in section 2.4, we apply the KLE+PCE surrogate to the mechanical oscillator example from chapter 1.

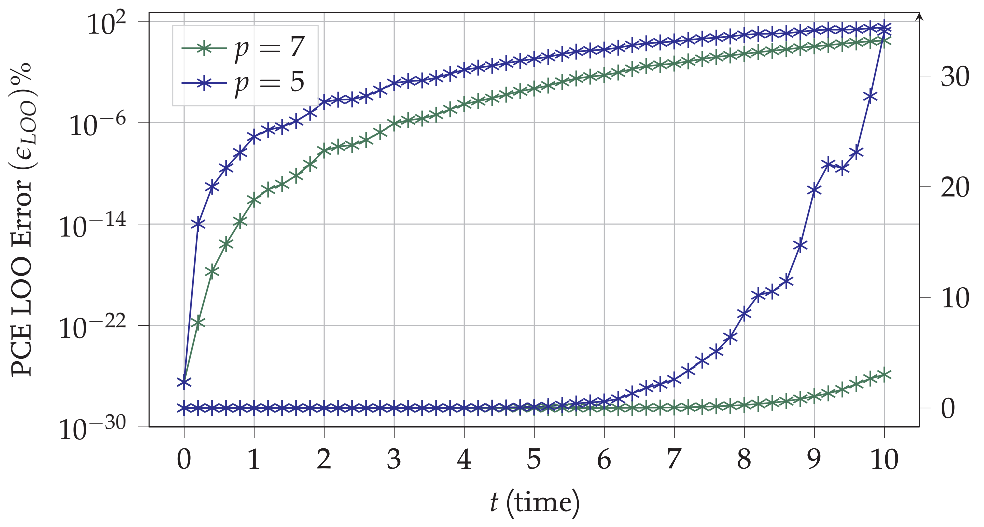


Figure 2.3.: Comparison of the Leave-One-Out error for the time frozen PCE surrogates of the mechanical oscillator for $p = 5$ (in blue) and $p = 7$ (in green). The left and right y-axes are plotted to illustrate the different scales.

2.2.1. Mathematical preliminaries

Consider a second-order random process $\mathcal{M}(t, \boldsymbol{\xi}(\omega)) : \mathcal{T} \times \Omega \to \mathbb{R}$ defined over a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ (refer equation 1.22). We assume[1] a centered random process,

$$\mathbb{E}[\mathcal{M}(t, \boldsymbol{\xi}(\omega))] = \int_{\Omega} \mathcal{M}(t, \boldsymbol{\xi}(\omega)) \, \mathrm{d}\mathbb{P}(\omega) = 0.$$

The covariance function of the process can be given as in (1.24),

$$\Gamma(t, s) := \operatorname{cov}(\mathcal{M}(t), \mathcal{M}(s)) = \mathbb{E}[\mathcal{M}(t, \cdot)\mathcal{M}(s, \cdot)], \quad t, s \in \mathcal{T}.$$

We can now define the covariance operator $\mathcal{C} : L^2(\mathcal{T}) \to L^2(\mathcal{T})$ of the process,

$$\mathcal{C}[u](s) := \int_{\mathcal{T}} \Gamma(t, s) u(t) \, \mathrm{d}t. \tag{2.37}$$

The operator is compact, symmetric, self-adjoint, has real non-negative eigenvalues $\{\lambda_i\}_{i=1}^{\infty}$ and each λ_i has linearly independent eigenvectors [19, 2].

The above properties allow us to apply *Mercer's theorem* which states that the orthonormal eigenfunctions $\{e_i\}_{i=1}^{\infty}$ of the covariance operator $\mathcal{C}$ form an orthonormal basis in $L^2(\mathcal{T})$ which can be used to decompose $\Gamma(t, s)$ as follows [2]:

$$\Gamma(t, s) = \sum_{i=1}^{\infty} \lambda_i e_i(t) e_i(s), \tag{2.38}$$

[1]In case the process is not centered, we can always use the relation (1.25) to center the process.

which is called the *spectral decomposition* of Γ. The eigenfunctions and eigenvalues are related as,

$$\mathcal{C}[e_i](s) = \int_{\mathcal{T}} \Gamma(t,s) e_i(t) \, \mathrm{d}t = \lambda_i e_i(s). \tag{2.39}$$

Here it is useful to draw parallels between functional analysis and linear algebra. Intuitively, the above linear operator $\mathcal{C}$ can be thought of as an infinite dimensional "matrix" (C with infinite elements) acting on an infinite dimensional "vector" u. The self-adjoint "matrix" $\mathcal{C}$ is equivalent to a Hermitian matrix (i.e. $\bar{C}^\top = C$) which has real and positive semi-definite eigenvalues. The Mercer's theorem is equivalent to saying $\mathcal{C}$ is diagonalisable, i.e. $C = V \Lambda V^{-1}$ and we know that for a Hermitian matrix, eigenvectors V form an orthogonal space (even when some eigenvalues are degenerate) [22].

We can now use the basis in (2.38) to decompose the random process,

$$\mathcal{M}(t,\boldsymbol{\xi}) = \sum_{i=1}^{\infty} \Phi_i(\boldsymbol{\xi}) e_i(t), \quad \Phi_i(\boldsymbol{\xi}) := \int_{\mathcal{T}} \mathcal{M}(t,\boldsymbol{\xi}) e_i(t) \, \mathrm{d}t, \tag{2.40}$$

where $\Phi_i(\boldsymbol{\xi})$ defined above is the projection of the random process on the orthonormal eigenbasis in $L^2(\mathcal{T})$. The error in a finite dimensional approximation of the random process with the first N_{kl} eigenfunctions can be given as follows [2]:

$$\epsilon_{N_{kl}} := \mathbb{E}\left[\left(\mathcal{M}(t,\boldsymbol{\xi}) - \sum_{i=1}^{N_{kl}} \Phi_i(\boldsymbol{\xi}) e_i(t)\right)^2\right] = \sum_{i=N_{kl}+1}^{\infty} \lambda_i, \tag{2.41}$$

which is simply the sum of the eigenvalues **not** considered in the approximation. The surrogate model is therefore,

$$\mathcal{M}(t,\boldsymbol{\xi}^{(j)}) \approx \sum_{i=1}^{N_{kl}} \Phi_i(\boldsymbol{\xi}^{(j)}) e_i(t). \tag{2.42}$$

2.2.2. Practical Computation

In practice, we rarely have closed form expressions for $\mathcal{M}(t,\boldsymbol{\xi})$. Usually only a finite discretisation of the random process is available, i.e. $\mathcal{M}(t_s,\boldsymbol{\xi}^{(j)})$ with $s \in \{1,...,m\}$ and $j \in \{1,...,N\}$. The compactness property of the covariance operator $\mathcal{C}$ allows us to approximate it in finite dimensions and build a finite dimensional basis. Therefore for most problems we resort to a finite dimensional approximation of the operator C^N, which is computed using N Monte Carlo samples of the random vector $\boldsymbol{\xi}$. We now outline the steps to compute the surrogate as in [6, 3],

1. Center the random process

We shall first center the random process using relation (1.25),

$$\mathcal{M}_c(t_s, \boldsymbol{\xi}^{(j)}) := \mathcal{M}(t_s, \boldsymbol{\xi}^{(j)}) - \frac{1}{N}\sum_{j=1}^{N} \mathcal{M}(t_s, \boldsymbol{\xi}^{(j)}), \quad s \in \{1, ..., m\}, \, j \in \{1, ..., N\}. \qquad (2.43)$$

where $\hat{\mu}_s = \frac{1}{N}\sum_{j=1}^{N} \mathcal{M}(t_s, \boldsymbol{\xi}^{(j)})$ is the mean of the process at $t = t_s$ as computed in (1.26).

Vector Notation

We denote the matrix containing the model evaluations by $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{N\times m}$, the mean of the process by $\boldsymbol{\mu_{\mathcal{Y}}} \in \mathbb{R}^{1\times m}$ and the centered process by the matrix $\boldsymbol{\mathcal{Y}}_c \in \mathbb{R}^{N\times m}$,

$$\boldsymbol{\mathcal{Y}} := \left\{ \boldsymbol{y}^{(1)} = \mathcal{M}\left(t_1, \boldsymbol{\xi}^{(1)}\right), \ldots, \boldsymbol{y}^{(N)} = \mathcal{M}\left(t_m, \boldsymbol{\xi}^{(N)}\right) \right\}^{\top}.$$

$$\boldsymbol{\mu_{\mathcal{Y}}} := \mathbb{E}[\boldsymbol{\mathcal{Y}}].$$

$$\boldsymbol{\mathcal{Y}}_c := \underbrace{\boldsymbol{\mathcal{Y}}}_{N\times m} .- \underbrace{\boldsymbol{\mu_{\mathcal{Y}}}}_{1\times m} .$$

where the operator $.-$ represents element-wise subtraction.

2. Compute the discretised covariance function

The covariance matrix $C^N \in \mathbb{R}^{m\times m}$ can now be simply computed using the above centered process as follows:

$$C^N_{pq} := \frac{1}{N-1}\sum_{j=1}^{N} \mathcal{M}_c(t_p, \boldsymbol{\xi}^{(j)})\mathcal{M}_c(t_q, \boldsymbol{\xi}^{(j)}), \quad p, q \in \{1, ..., m\}. \qquad (2.44)$$

Vector Notation

$$C^N := \frac{1}{N-1} \underbrace{\boldsymbol{\mathcal{Y}}_c^{\top}}_{m\times N} \underbrace{\boldsymbol{\mathcal{Y}}_c}_{N\times m} .$$

3. Eigendecomposition of C^N

In order to compute the orthogonal basis of C^N we discretise equation (2.39) and solve it using a quadrature. Since typically a solution with a uniform discretisation is available, we choose the trapezoidal quadrature with weights $\{w_q\}_{q=1}^{m}$ and nodes $\{t_q\}_{q=1}^{m}$ [3],

$$\sum_{q=1}^{m} w_q C^N(s_p, t_q) e_i(t_q) = \lambda_i e_i(s_p), \quad i \in \{1, ..., m\}, \quad p \in \{1, ..., m\}. \qquad (2.45)$$

We re-write the above eigenvalue problem in the matrix form as follows:

Vector Notation

The weights are stored in a diagonal matrix $\boldsymbol{W} = \operatorname{diag}(w_1, \ldots, w_m)$ and the generalised eigenvalue problem is:

$$\boldsymbol{K}\boldsymbol{W}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i, \quad i \in \{1, \ldots, m\}. \tag{2.46}$$

Equivalently,

$$\boldsymbol{W}^{1/2}\boldsymbol{K}\boldsymbol{W}^{1/2}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i, \quad i \in \{1, \ldots, m\}, \tag{2.47}$$

where $\boldsymbol{v}_i = \boldsymbol{W}^{-1/2}\boldsymbol{u}_i \in \mathbb{R}^m$ is the eigenvector and $\lambda_i \in \mathbb{R}$ is the corresponding eigenvalue.

We denote the matrix containing all eigenvectors by $\boldsymbol{V} \in \mathbb{R}^{m\times m}$ and the diagonal matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{m\times m}$ contains the eigenvalues in descending order. Since we use the trapezoidal rule, $\boldsymbol{W} = \operatorname{diag}(\frac{1}{2}, 1, \ldots, 1, \frac{1}{2})$.

4. Choose truncation order

For some processes the magnitude of eigenvalues decreases sharply and a small number of eigenvectors are sufficient to approximate the process. Such processes are termed *low-rank*. The truncation level N_{kl} is based on the point-wise error estimate in (2.42). Practically we choose a truncation level based of the explained variance,

$$r_{N_{kl}} := \frac{\sum_{i=1}^{N_{kl}} \lambda_i}{\sum_{i=1}^{m} \lambda_i}, \tag{2.48}$$

where $N_{kl} \ll m$ for low-rank processes. We denote the truncated eigenvector space by $\boldsymbol{V}_{N_{kl}} \in \mathbb{R}^{m\times N_{kl}}$

5. Compute projections

Once the truncation order has been chosen, we can now compute $\Phi_i(\boldsymbol{\xi})$ in equation (2.40). In the discretised case, we obtain the value of the projections for various realisation of $\boldsymbol{\xi}$ using a quadrature as mentioned above,

$$\Phi_i(\boldsymbol{\xi}^{(j)}) := \sum_{s=1}^{m} w_s \mathcal{M}_c(t_s, \boldsymbol{\xi}^{(j)}) v_i(t_s), \quad i \in \{1, \ldots, N_{kl}\}, j \in \{1, \ldots, N\}. \tag{2.49}$$

Vector Notation

Here the matrix $\Phi \in \mathbb{R}^{N \times N_{kl}}$ denotes the projections of the centered process $\boldsymbol{\mathcal{Y}}_c$ on the truncated eigenbasis $\boldsymbol{V}_{N_{kl}}$ with quadrature weights in $\boldsymbol{W}$.

$$\Phi := \underbrace{\boldsymbol{\mathcal{Y}}_c}_{N \times m} \underbrace{\boldsymbol{W}}_{m \times m} \underbrace{\boldsymbol{V}_{N_{kl}}}_{m \times N_{kl}} .$$

Clearly each column of the matrix Φ (denoted by Φ_i above) contains evaluation of the projection for N realisations of $\boldsymbol{\xi}$, which we shall refer to as *mode* as in [3].

6. Approximate Projections with PCE

We now construct a PCE surrogate for each mode Φ_i described above, so that we can evaluate the surrogate for new realisations $\boldsymbol{\xi}$ at various time-points. For construction of gPCE surrogates refer section 2.1. We denote the gPCE approximation of the modes as follows:

$$\Phi_i(\boldsymbol{\xi}) \approx \tilde{\Phi}_i^{PC}(\boldsymbol{\xi}) = \sum_{k=1}^{N_{PC,\Phi}} f_{\Phi_i}^{(k)} \mathcal{F}_{\Phi_i}^{(k)}(\boldsymbol{\xi}),$$

where $N_{PC,\Phi}$ is the number of terms in the polynomial basis (generally the same value is chosen for all modes), $f_{\Phi_i}^{(k)}$ are the basis coefficients and $\mathcal{F}_k(\boldsymbol{\xi})$ represents the multivariate orthonormal polynomial basis for the i^{th} mode.

Surrogate evaluation

We begin by evaluating a gPCE surrogate $\tilde{\Phi}_i^{PC}(\boldsymbol{\xi})$ for the i^{th} mode for a realisation of the input vector $\boldsymbol{\xi}^{(j)}$ and sum over the truncated eigenvectors. In step 1, we centered the process in order to simplify the computation of the covariance estimate C^N. We must now therefore add the mean back,

$$\hat{\mathcal{M}}(t_s, \boldsymbol{\xi}^{(j)}) \approx \hat{\mu}_s + \sum_{i=1}^{N_{kl}} \underbrace{\sum_{k=1}^{N_{PC,\Phi}} f_{\Phi_i}^{(k)} \mathcal{F}_{\Phi_i}^{(k)}(\boldsymbol{\xi}^{(j)})}_{\kappa_{ij}} v_i(t_s), \quad s \in \{1, \ldots, \hat{m}\}, \, j \in \{1, \ldots, \hat{N}\}, \tag{2.50}$$

where κ_{ij} is a term in the Vandermonde matrix as defined in (2.29). Note that the surrogate can be evaluated at a subset of the original time discretisation for efficiency if necessary (denoted by $\hat{m}$).

Vector Notation

In order to evaluate the surrogate, we first compute the Vandermonde matrix $\mathcal{K} \in \mathbb{R}^{\hat{N} \times N_{kl}}$ and project it on the truncated eigenbasis $\boldsymbol{V}_{N_{kl}}^{\top}$ and add back the mean

$$\hat{\mathcal{Y}} = \underbrace{\mathcal{K}}_{\hat{N} \times N_{kl}} \underbrace{\boldsymbol{V}_{N_{kl}}^{\top}}_{N_{kl} \times \hat{m}} .+ \underbrace{\boldsymbol{\mu}_{\mathcal{Y}}}_{1 \times \hat{m}},$$

where the operator $.+$ represents element-wise addition.

2.2.3. Error estimate

The errors in the KLE+PCE surrogate has three components as outlined in [6],

1. KLE Truncation Error ($\epsilon_{N_{kl}}$): Error due to the truncated KLE basis is given in equation (2.41). But we can only estimate this error since we compute a finite number of bases using the covariance matrix. We divide it by the total variance to obtain a relative error estimate.

$$\epsilon_{N_{kl}} = \frac{\sum_{i=N_{kl}+1}^{\infty} \lambda_i}{\sum_{i=1}^{\infty} \lambda_i} \approx \frac{\sum_{i=N_{kl}+1}^{m} \lambda_i}{\sum_{i=1}^{m} \lambda_i}.$$

2. Sampling Error (ϵ_N): Error in the covariance matrix due to the estimate with finite samples in equation (2.44).
3. PCE approximation Error ($\epsilon_{LOO,KL}$): The modes are approximated using gPCE surrogates and thus we can estimate the relative error using the Leave-One-Out cross-validation method summarised in equation 2.34 for each mode ($i \in \{1, ..., N_{kl}\}$). The total error $\epsilon_{LOO,KL}$ can be computed as follows:

$$\epsilon_{LOO,KL} \approx \frac{\sum_{i=1}^{N_{kl}} \epsilon_{LOO,i}}{\sum_{i=1}^{m} \lambda_i}.$$

The quality of the approximation can be measured by combining $\epsilon_{N_{kl}}$ and $\epsilon_{LOO,KL}$ as follows [proposed in 6]:

$$\epsilon_{KL+PC} := \left(\sqrt{\epsilon_{N_{kl}}} + \sqrt{\epsilon_{LOO,KL}}\right)^2. \tag{2.51}$$

2.2.4. Example: Mechanical Oscillator

In this section we shall compute the KLE+PCE surrogate for the mechanical oscillator example (1.21) from section 1. Unlike the local time-frozen surrogates in section 2.1.6, here we shall construct a global surrogate for the time-dependent model $\mathcal{M}(t, \xi)$.

For the surrogate, we choose $N = 20,000$ Monte Carlo samples to estimate the covariance matrix and choose $m = 101$ discretisations in $\mathcal{T} = [0, 10]$. Since a closed form solution is available for this random process, it is computationally inexpensive to compute the model evaluations matrix $\mathcal{Y}$. The random process with 50 realisations is plotted in Figure 1.5 and the centered model evaluations have been illustrated in Figure 1.7.

We now plot the eigenvalues of the covariance matrix C^N in Figure 2.4. The quick decrease in the eigenvalues implies that the process is low-rank. We therefore choose a truncation level $N_{kl} = 6$ which leads to $r_{N_{kl}} = 0.9981$ which means that the first 6 eigenvectors are sufficient to explain 99.81% of the total variance and therefore the process can be approximated fairly well using just $N_{kl} = 6$ basis vectors. To illustrate the new eigenbasis, we plot the first 4 eigenvectors in Figure 2.7. Although these are eigenvectors (i.e discretised versions of the eigenfunctions), the fine time discretisation $m = 101$ make the plots look continuous. In Figure 2.7, we also superimpose the first realisation of the centered process $\mathcal{M}_c(t, \boldsymbol{\xi}^{(1)})$ in order to elucidate the dot product in the projection $< \mathcal{M}_c(t, \boldsymbol{\xi}^{(1)}), e_i(t) >$ (2.49) for the first 4 eigenvectors.

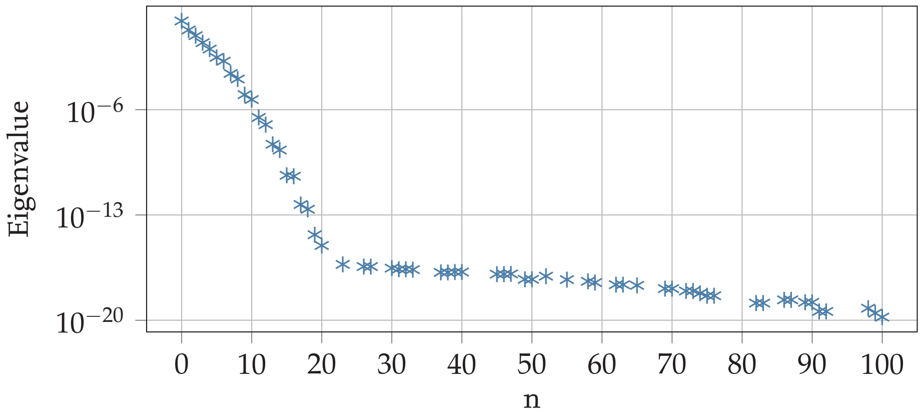


Figure 2.4.: Eigenvalues of the covariance operator of the mechanical oscillator example computed with $N = 20,000$ Monte Carlo samples. The quick drop in the eigenvalues suggests that the underlying process is low-rank.

To understand the effect of the variance-based truncation, we plot the empirical point-wise variance of process in Figure 2.5, which is computed as,

$$\mathbb{V}[\mathcal{M}_c(t_s, \boldsymbol{\xi})] \approx \frac{1}{N-1} \sum_{j=1}^{N} \mathcal{M}_c(t_s, \boldsymbol{\xi}^{(j)})^2, \quad s \in \{1, \ldots, m\}.$$

Similarly we can compute the point-wise variance of the response of the surrogate $\tilde{\mathcal{M}}_c(t_s, \boldsymbol{\xi})$. The relative error plotted in Figure 2.5 (in red) is computed as follows:

$$\frac{\mathbb{V}[\mathcal{M}(t_s, \boldsymbol{\xi})] - \mathbb{V}[\tilde{\mathcal{M}}^{(KL+PC)}(t_s, \boldsymbol{\xi})]}{\mathbb{V}[\mathcal{M}(t, \boldsymbol{\xi})]} \times 100, \quad s \in \{1, \ldots, m\}.$$

We see that the relative error (in percentage) increases in the area of low variance and this can be explained as follows: since we use a variance-based truncation, the KLE approximation tries to capture areas of large variance and areas with low variance are therefore neglected since they make a small contribution to the total variance.

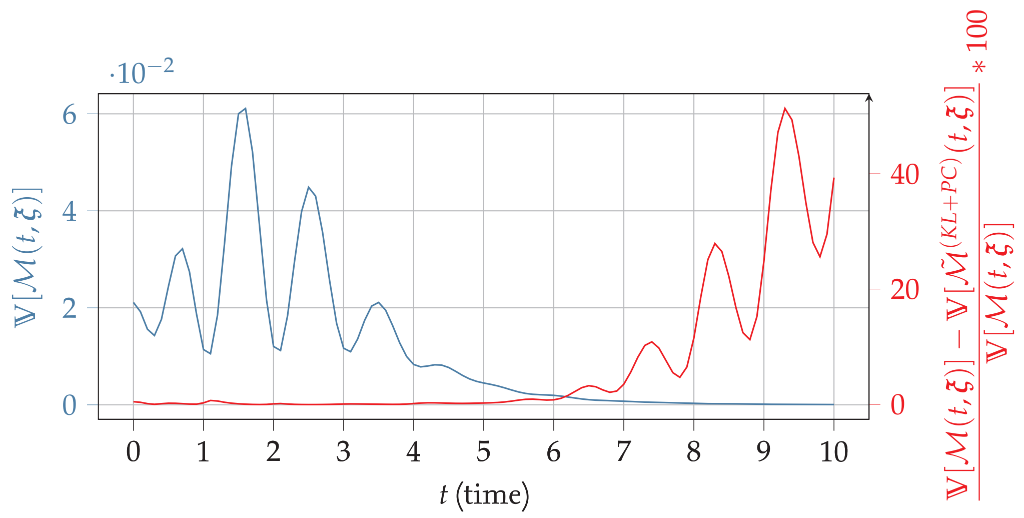


Figure 2.5.: Point-wise variance in the mechanical oscillator example (in blue) decreases with time. Since we use a variance-based truncation for the basis, areas with low variance are neglected and a high error in approximating their variance is proven by the point-wise relative error curve (in red).

Lastly, we compute the error $\epsilon_{LOO,i}$ in approximating the modes using gPCE. The Leave-One-Out error for each mode is plotted in Figure 2.6. Here all gPCE approximations are computed using Legendre polynomials (since the random variables are modeled using uniform random variables) of degree 5 and therefore each approximation has an orthogonal polynomial basis with $N_{PC} = \frac{(N_p+p)!}{N_p!p!} = \frac{(3+5)!}{3!5!} = 56$ terms. The increasing error in the gPCE approximated modes implies that higher modes are relatively more intricate and require a larger basis for an accurate approximation, which agrees with the observation in [6].

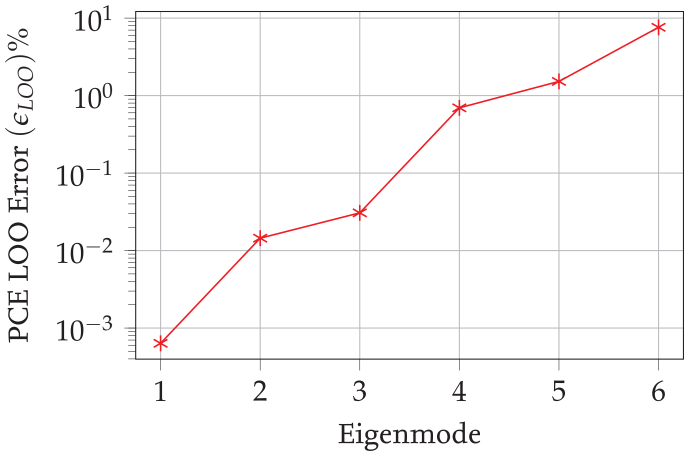


Figure 2.6.: Leave-One-Out error for the projections $\Phi_i(\xi)$ approximated using gPCE.

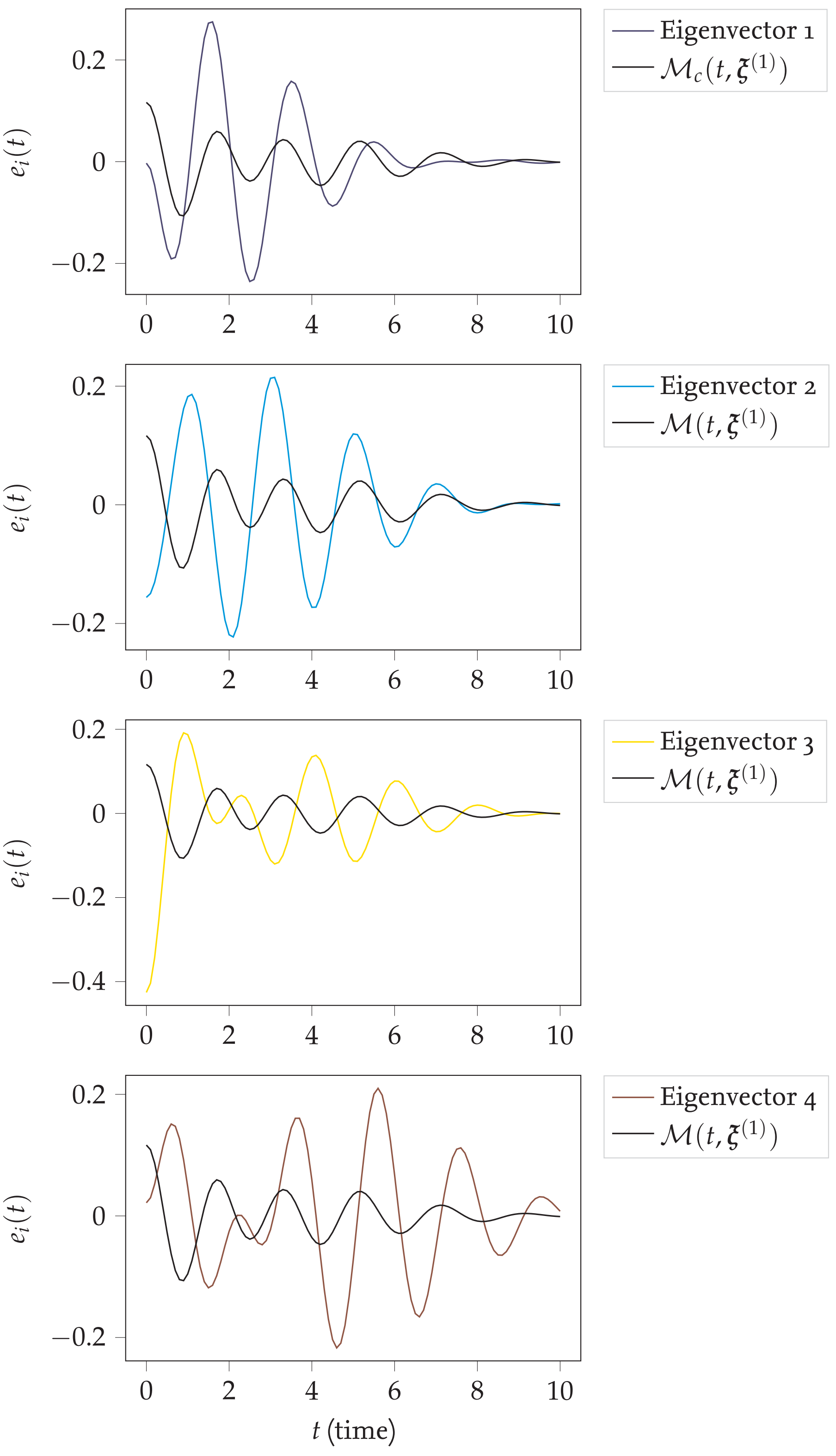


Figure 2.7.: Top to bottom: First four eigenvectors of the KLE expansion of the mechanical oscillator random process. The first centered realisation of the random process (in black) is plotted in each graph to illustrate the of the "projection" of the centered process on the eigenbasis.

3. Global Sensitivity Analysis

Chapter 1 illustrated the forward propagation of uncertainties arising from uncertain model inputs. It is now natural to ask how the uncertainties in the model output can be "apportioned" or "allocated" to the uncertainties in the model inputs [24]. Intuitively speaking, we would like to quantify the contribution of uncertainty stemming from each random input variable towards the model output uncertainty. This task in UQ is called *Sensitivity Analysis* (SA) and is typically carried out in conjunction with forward UQ. Here are some of the common goals for SA studies [24],

- **Factor Prioritisation**: SA can tell us which input variables make large contributions to the output uncertainty. Once these factors are identified, steps can be taken to reduce uncertainty in those input variables.
- **Factor Fixing**: Similarly, it is possible to identify which input variables have the least contribution to the output uncertainty. This enables us to set such parameters to a nominal value and simplify the model.
- **Experiment Design**: Certain input variables of a model may have a large contribution to the output uncertainty in certain settings. SA can help us identify optimal experimental settings under which a variable is most responsive in order to calibrate it.

Sensitivities of a model can be studied in two ways, *Local Sensitivity Analysis* (LSA) and *Global Sensitivity Analysis* (GSA) and there are several metrics available within each category (an overview can be found in [9]). LSA studies the response of the model to a local perturbation in the input space. In the problem settings considered in this work, this method has two limitations [24],

- Since the model inputs are random variables, a deterministic point-wise perturbation of the input space is not meaningful.
- LSA can only provide local information about the model sensitivities.

In order to address the above limitations, we resort to GSA, which allows us to study the sensitivities of the model response w.r.t the entire stochastic input space. Within the regime of GSA, there are several ways to compute global sensitivities such as regression based methods, variance-based methods (Sobol indices), density-based methods (Borgonovo indices), Monte Carlo filtering etc. [9]. In this work, we shall use variance-based methods to compute sensitivities for two reasons: they are robust and can be applied to any model with finite variance and secondly, they can capture interaction effects of model inputs. Variance-based metrics however, as we shall see, have a high computational cost (these can be alleviated to a reasonable degree by adopting surrogate modeling techniques as introduced in chapter 2). We illustrate the essence of variance-based sensitivity analysis using the following example,

Example 3.1: Consider a trivial model $Z(\xi_1, \xi_2) := 3\xi_1 + \xi_2$, where $\xi_1 \sim \mathcal{N}(0,1)$ and $\xi_2 \sim \mathcal{U}(-6,6)$. Clearly, $\mathbb{E}[\xi_1] = 0$, $\mathbb{E}[\xi_2] = 0$ (eq. (1.14)) and $\mathbb{V}[\xi_1] = 1$, $\mathbb{V}[\xi_2] = \frac{36^2}{12} = 108$ (eq. (1.15)). Using equation (1.10), the total variance of the output can be computed as follows:

$$\mathbb{V}[Z] = \mathbb{V}[3\xi_1 + \xi_2] = 9\mathbb{V}[\xi_1] + \mathbb{V}[\xi_2] = 9 \cdot 1 + 1 \cdot 108.$$

The fraction of variance induced by the first and second variable are denoted by $\mathcal{S}_1 := \frac{9}{9+108} \approx 0.08$ and $\mathcal{S}_2 := \frac{108}{9+108} \approx 0.92$ respectively. This implies that parameter ξ_1 is responsible for 8% of the model uncertainty and the remaining 92% uncertainty is caused by ξ_2. Clearly, sensitivities are a relative measure and depend on the uncertainty in the input variable. This implies that a decrease in the uncertainty in a parameter would decrease its sensitivity. This example however does not illustrate interaction effects of the parameters and this technique is limited to simple models. To tackle problems of practical importance, we use a generalisation of this idea and use numerical techniques to compute sensitivities.

In this chapter, we proceed as follows: we introduce the variance-based sensitivity indices for models with scalar outputs called *Sobol indices* (SI) in section 3.1. For models with a vector-valued response we introduce the *Generalised Sobol Indices* (GSI) in section 3.2. In both sections we shall discuss numerically efficient techniques to compute the indices and illustrate their use through the mechanical oscillator example from chapter 1.

3.1. Sobol Indices

Variance-based global sensitivities are ubiquitously quantified using Sobol indices and have become synonyms since their introduction by the Russian mathematician I.M.Sobol [26]. In this section we shall review the mathematical foundations of the indices and look at algorithms for their practical computation.

3.1.1. Mathematical preliminaries

Consider a square integrable model $Y = \mathcal{M}(\boldsymbol{\xi}) : \Omega \to \mathbb{R}$ with independent random input variables $\boldsymbol{\xi} \in \Omega = [0,1]^{N_p}$. Although we require that $\xi_i \sim \mathcal{U}(0,1)$, it is only a mild assumption. We can always transform variables $\boldsymbol{X} \sim \mathcal{D}$ belonging to general distribution onto $\mathcal{U}(0,1)$ using some transformation $T(\boldsymbol{\xi}) = \boldsymbol{X}$, where $T(\boldsymbol{\xi})$ could, for example, be a PCE (as introduced in section 2.1).

Sobol Decomposition

If the above assumptions hold, the *Sobol Decomposition* of $\mathcal{M}(\boldsymbol{\xi})$ is defined as follows:

$$Y = \mathcal{M}(\boldsymbol{\xi}) = Y_0 + \sum_{i=1}^{N_p} Y_i(\xi_i) + \sum_{i=1}^{N_p}\sum_{j>i}^{N_p} Y_{ij}(\xi_i, \xi_j) + \ldots + Y_{1,\ldots,N_p}(\xi_1, \ldots, \xi_{N_p}). \tag{3.1}$$

The above decomposition, represented as a sum of all possible combinations of variable interactions, is called a *High Dimensional Model Representation* (HDMR). The decomposition contains 2^{N_p} terms and can be computed as follows:

$$\begin{aligned} Y_0 &= \mathbb{E}[\mathcal{M}(\boldsymbol{\xi})], \\ Y_i &= \mathbb{E}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i] - Y_0, \\ Y_{ij} &= \mathbb{E}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i, \xi_j] - Y_i - Y_j - Y_0, \end{aligned}$$

The other terms in the decomposition can be computed similarly. The above representation (3.1) is not unique but has the following useful properties,

1. Each term in the expansion has zero mean: $\int_{[0,1]^{N_p}} Y_A(\boldsymbol{\xi}_A)\, \mathrm{d}\boldsymbol{\xi}_A = 0$.
2. Terms are mutually orthogonal:$\int_{[0,1]^{N_p}} \int_{[0,1]^{N_p}} Y_A(\boldsymbol{\xi}_A) Y_B(\boldsymbol{\xi}_B)\, \mathrm{d}\boldsymbol{\xi}_A\, \mathrm{d}\boldsymbol{\xi}_B = 0$,

where A, B represent ordered subsets of $\{1, \ldots, N_p\}$ and $Y_A(\boldsymbol{\xi}_A), Y_B(\boldsymbol{\xi}_B)$ represent terms from the decomposition (3.1).

Remark: Although variance is an intuitive choice to convey uncertainty in the model output, it is known that moments alone cannot completely convey the uncertainty. Therefore relying solely on variance as an uncertainty metric can be misleading [9]. For an academic example of such a case refer [8].

First-Order Sobol Indices

The notion of Sobol indices is illustrated in example 3.1 for a simple linear additive model. However most models in practice have non-linearities and interactions of input variables. For such general non-additive models, the following generalisation can be used: the sensitivity of an input variable is determined by the reduction in the variance of the model output caused by the fixing the value of the variable [9]. The generalisation can be mathematically formulated as follows [24]:

$$\mathcal{S}_i := \frac{\mathbb{V}_{\xi_i}[\mathbb{E}_{\boldsymbol{\xi}|\xi_i}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i]]}{\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})]}, \tag{3.2}$$

where $\mathcal{S}_i$ denotes the *First-Order Sobol Index* of the variable ξ_i. The term $\mathbb{E}_{\boldsymbol{\xi}|\xi_i}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i]$ in the above expression refers to the expectation of the model response over all parameters, except ξ_i, after fixing the value of the variable $\xi_i = \xi_i^*$. Now in order to make this quantity independent of the chosen value ξ_i^*, we must compute the variance of the term for various values of ξ_i, i.e $\mathbb{V}_{\xi_i}[\cdot]$. To make this a relative measure, we divide by the total variance of the model $\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})]$ so that $0 \leq \mathcal{S}_i \leq 1$. For a more comprehensive explanation of the terms refer [p.20, 24].

Total-Order Sobol Indices

Similar to the relation (3.2), we can define second order Sobol indices as follows:

$$\mathcal{S}_{ij} := \frac{\mathbb{V}_{\xi_{ij}}[\mathbb{E}_{\boldsymbol{\xi}|\xi_i,\xi_j}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i, \xi_j]]}{\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})]}. \tag{3.3}$$

One can continue to define such indices for all $2^{N_p} - 1$ terms in the expansion (3.1). But the cost of computing $2^{N_p} - 1$ indices can be too large and therefore we define the *Total-Order Sobol Index* $\mathcal{S}_{T_i}$ to capture the total influence of the variable ξ_i on the model output. This term includes the first-order and interaction effects of the variable. The total order indices are defined as follows:

$$\mathcal{S}_{T_i} := 1 - \frac{\mathbb{V}_{\boldsymbol{\xi}|\xi_i}[\mathbb{E}_{\xi_i}[\mathcal{M}(\boldsymbol{\xi}) \mid \boldsymbol{\xi} \mid \xi_i]]}{\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})]}. \tag{3.4}$$

The explanation of the above terms are analogous to terms in (3.2). The fraction essentially computes the contribution of all the terms, except the terms containing the variable ξ_i, towards the output variance. Subtracting the fraction from 1 tells us the total contribution of all terms containing ξ_i towards the total variance. In order to compute the interaction effects of a variable, we can simply subtract the First-order effect from the Total-order effect,i.e. $\mathcal{S}_{T_i} - \mathcal{S}_i$. When computing only the First-order and Total-order indices, we compute $2N_p$ indices instead of $2^{N_p} - 1$, which is significantly more efficient.

ANOVA Decomposition

The Sobol decomposition in (3.1) enables us to compute the Sobol indices very efficiently using the orthogonality of the terms. Upon applying the variance operator on both sides of the decomposition and divding both sides by $\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})]$ we obtain the *Analysis of Variance* (AVONA) decomposition,

$$1 = \sum_{i=1}^{N_p} \mathcal{S}_i + \sum_{i=1}^{N_p} \sum_{j>i}^{N_p} \mathcal{S}_{ij} + \ldots + \mathcal{S}_{1,\ldots,N_p}. \tag{3.5}$$

Example 3.1.1 A model $\mathcal{M}(\xi_1, \xi_2, \xi_3)$ with $N_p = 3$ random variable has $2^3 = 8$ terms in the decomposition and are given as follows,

$$\begin{aligned}\mathcal{M}(\xi_1, \xi_2, \xi_3) =& Y_0 + \\ & Y_1(\xi_1) + Y_2(\xi_2) + Y_3(\xi_3) + \\ & Y_{12}(\xi_1, \xi_2) + Y_{13}(\xi_1, \xi_3) + Y_{23}(\xi_2, \xi_3) + \\ & Y_{123}(\xi_1, \xi_2, \xi_3).\end{aligned}$$

The First-order Sobol indices and Total-order Sobol indices of ξ_1, ξ_2, ξ_3 are given by $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ and $\mathcal{S}_{T_1}, \mathcal{S}_{T_2}, \mathcal{S}_{T_3}$ respectively. The interaction effects of the variable ξ_1 is given by $\mathcal{S}_{T_1} - \mathcal{S}_1 = \mathcal{S}_{12} + \mathcal{S}_{13} + \mathcal{S}_{123} = \underbrace{1 - \mathcal{S}_2 - \mathcal{S}_3 - \mathcal{S}_{23}}_{\mathcal{S}_{T_1}} - \mathcal{S}_1$.

3.1.2. Practical Computation

Pick and Freeze Estimator

The *Pick and Freeze Estimator* is a Monte Carlo estimator which can be used to compute the First-order and Total-order Sobol indices of a model. We begin by generating $2N$ Monte Carlo samples of the input variables and storing them in matrices $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{N \times N_p}$,

$$\boldsymbol{\mathcal{A}} := \left\{ \boldsymbol{\xi}_A^{(1)}, \dots, \boldsymbol{\xi}_A^{(N)} \right\}^\top, \quad \boldsymbol{\mathcal{B}} := \left\{ \boldsymbol{\xi}_B^{(1)}, \dots, \boldsymbol{\xi}_B^{(N)} \right\}^\top. \tag{3.6}$$

We create another input matrix $\boldsymbol{\mathcal{C}}_i, i \in \{1, \dots, N_p\}$ which contains all columns of matrix $\boldsymbol{\mathcal{B}}$ except column i, which contains entries of column i of matrix $\boldsymbol{\mathcal{A}}$. The model is evaluated for all entries of matrices $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}$ and $\boldsymbol{\mathcal{C}}_i$ and stored in vectors $\mathcal{Y}_\mathcal{A}, \mathcal{Y}_\mathcal{B}, \mathcal{Y}_{\mathcal{C}_i} \in \mathbb{R}^N$ respectively. For example, the elements of vector $\mathcal{Y}_\mathcal{A}$ are computed as follows:

$$\mathcal{Y}_\mathcal{A} = \left\{ y_A^{(1)} = \mathcal{M}\left(\boldsymbol{\xi}_A^{(1)}\right), \dots, y_A^{(N)} = \mathcal{M}\left(\boldsymbol{\xi}_A^{(N)}\right) \right\}^\top. \tag{3.7}$$

In order to efficiently compute the sensitivities, we must first estimate the mean $\hat{\mu}$ and total variance $\hat{V}$. We use the model evaluations $\mathcal{Y}_\mathcal{A}$ to estimate these values as follows:

$$\mathbb{E}[\mathcal{M}(\boldsymbol{\xi})] \approx \hat{\mu}_A = \frac{1}{N} \sum_{j=1}^{N} y_A^{(j)}. \tag{3.8}$$

$$\mathbb{V}[\mathcal{M}(\boldsymbol{\xi})] = \mathbb{E}[\mathcal{M}^2(\boldsymbol{\xi})] - \mathbb{E}[\mathcal{M}(\boldsymbol{\xi})]^2 \approx \hat{V} = \frac{1}{N-1} \sum_{i=1}^{N} y_A^{(j)} y_A^{(j)} - \hat{\mu}^2. \tag{3.9}$$

The First-order and Total-order Sobol indices for a random variable ξ_i are given in equation (3.10) and (3.11) respectively [p.164 24].

$$\mathcal{S}_i = \frac{\frac{1}{N} \mathcal{Y}_\mathcal{A}^\top \mathcal{Y}_{\mathcal{C}_i} - \hat{\mu}^2}{\hat{V}} = \frac{\frac{1}{N} \sum_{j=1}^{N} y_A^{(j)} y_{C_i}^{(j)} - \hat{\mu}^2}{\hat{V}}. \tag{3.10}$$

$$\mathcal{S}_{T_i} = 1 - \frac{\frac{1}{N}\mathcal{Y}_{\mathcal{B}}^{\top}\mathcal{Y}_{C_i} - \hat{\mu}^2}{\hat{V}} = 1 - \frac{\frac{1}{N}\sum_{j=1}^{N} y_{B}^{(j)} y_{C_i}^{(j)} - \hat{\mu}^2}{\hat{V}}. \tag{3.11}$$

Since we have N sets of inputs from each matrix $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}$ and N sets of inputs from each of the N_p $\boldsymbol{\mathcal{C}}_i$ matrices, the computations of these indices requires $N(N_p + 2)$ model evaluations.

Using PCE coefficients

It has been shown in [28] that the PCE representation (2.18) is equivalent to the HDMR representation in (3.1) and in the same work, the authors demonstrate how the PCE coefficients can be directly used to compute Sobol Indices of any order at negligible cost . This has two implications, firstly the cost of computing indices changes from $N(N_p + 2)$ to the cost of constructing a PCE surrogate, secondly a surrogate model now becomes available for other UQ tasks.

We begin by computing the mean and variance of the model using the PCE surrogate. Using the orthogonality of the polynomial basis and independence of random variables we obtain,

$$\mathbb{E}\left[\tilde{\mathcal{M}}^{PC}(\boldsymbol{\xi})\right] = \mathbb{E}\left[\sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi})\right] = c_1 + \sum_{k=2}^{N_{PC}} c_k \mathbb{E}\left[\Psi_k(\boldsymbol{\xi})\right], \tag{3.12}$$

$$= c_1 + \sum_{k=2}^{N_{PC}} c_k \prod_{i=1}^{N_p} \mathbb{E}\left[\psi_{\alpha_i^k}^{(i)}(\xi_i)\right], \quad \text{using (2.12)} \tag{3.13}$$

$$= c_1 + \sum_{k=2}^{N_{PC}} c_k \prod_{i=1}^{N_p} \underbrace{\mathbb{E}\left[\psi_{\alpha_i}^{(i)}(\xi_i) \cdot 1\right]}_{=0}, \quad \text{using (2.7)} \tag{3.14}$$

$$= c_1. \tag{3.15}$$

Similarly, using the orthogonality of the basis, we can compute the variance of the model using the PCE surrogate, which simply the sum of square of the coefficients minus the square of the mean.

$$\mathbb{V}\left[\tilde{\mathcal{M}}^{PC}(\boldsymbol{\xi})\right] = \mathbb{E}\left[\left(\sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi})\right)^2\right] - c_1^2, = \mathbb{E}\left[\sum_{k=1}^{N_{PC}}\sum_{j=1}^{N_{PC}} c_k c_j \Psi_k(\boldsymbol{\xi})\Psi_j(\boldsymbol{\xi})\right] - c_1^2,$$

$$= \sum_{k=1}^{N_{PC}}\sum_{j=1}^{N_{PC}} c_k c_j \underbrace{\mathbb{E}\left[\Psi_k(\boldsymbol{\xi})\Psi_j(\boldsymbol{\xi})\right]}_{\delta_{jk}} - c_1^2, \quad \text{using (2.13)} \tag{3.16}$$

$$= \sum_{k=2}^{N_{PC}} c_k^2.$$

The notation for deriving the Sobol indices is fairly cumbersome and therefore we shall provide an intuitive explanation for obtaining the coefficients and support it with an example.

Consider the term $\mathbb{E}_{\boldsymbol{\xi}|\xi_i}[\mathcal{M}(\boldsymbol{\xi}) \mid \xi_i]$ in (3.2). For a PCE surrogate we can re-write it as follows using relations (2.18) and (2.12),

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\xi}|\xi_i}[\tilde{\mathcal{M}}^{PC}(\boldsymbol{\xi}) \mid \xi_i] &= \mathbb{E}_{\boldsymbol{\xi}|\xi_i}\left[\sum_{k=1}^{N_{PC}} c_k \Psi_{\boldsymbol{\alpha}^k}(\boldsymbol{\xi}) \mid \xi_i\right], \text{using (2.18)}\\ &= \mathbb{E}_{\boldsymbol{\xi}|\xi_i}\left[\sum_{k=1}^{N_{PC}} c_k \prod_{j=1}^{N_p} \psi^{(j)}_{\alpha^k_j}(\xi_j) \mid \xi_i\right], \text{using (2.12)}\end{aligned}$$

Using the linearity of the expectation operator and independence of random variables we can focus on the following term,

$$\mathbb{E}_{\boldsymbol{\xi}|\xi_i}\left[\psi^{(j)}_{\alpha^k_j}(\xi_j) \mid \xi_i\right].$$

We know from (3.13) that the expectation of the polynomial terms is zero. Therefore the coefficients c_k of the terms $\psi^{(j)}_{\alpha^k_j}(\xi_j)$ with $\alpha_j \neq 0, j \neq i$ will vanish. Alternatively, only the terms $\psi^{(j)}_{\alpha^k_j}(\xi_j)$ with $\alpha_j = 0, j \neq i$ will remain. Intuitively, all the coefficients of univariate terms other than ξ_i and all the mixed terms will vanish and all the coefficients of the monomials of ξ_i will remain. As per (3.2), we must compute the variance of the "remaining" terms, which is simply the square of these coefficients (3.16) (since their mean is zero). The denominator in (3.2) is simply the total variance, which can be computed using (3.16).

Similarly, the Total-order Sobol index of ξ_i can be computed by squaring coefficients of all terms containing the term ξ_i, i.e. all monomials and all the mixed terms containing ξ_i. The coefficients to be chosen to compute the First-order and Total-order Sobol indices can be summarised using as follows:

Coefficient pickers $\mathcal{K}$

First-order coefficients picker

$$\mathcal{K}_i := \left\{k \in \{1, \ldots, N_{PC}\} \,\middle|\, \psi^{(j)}_{\alpha^k_j}(\xi_j),\, \forall j,\, \alpha^k_j = 0, j \neq i\right\}, \quad i = \{1, \ldots, N_p\}. \quad (3.17)$$

Total-order coefficients picker

$$\mathcal{K}_{T_i} := \left\{k \in \{1, \ldots, N_{PC}\} \,\middle|\, \psi^{(j)}_{\alpha^k_j}(\xi_j),\, \forall j,\, \alpha^k_i \neq 0\right\}, \quad i = \{1, \ldots, N_p\}. \quad (3.18)$$

Example 3.2: Consider a model $\mathcal{M}(\xi_1, \xi_2, \xi_3)$ with $N_p = 3$ random variables with $\xi_i \sim \mathcal{U}(0,1)$. For simplicity, we construct a PCE surrogate with $p = 2$, which has $N_{PC} = \frac{(N_p+p)!}{N_p!p!} = \frac{(3+2)!}{3!2!} = 10$ terms. Using Legendre orthogonal polynomials in (2.10), the PCE approximation can be given as follows,

$$\tilde{\mathcal{M}}^{PC}(\xi_1,\xi_2,\xi_3) = c_1 + c_2\xi_1 + c_3\xi_2 + c_4\xi_3 + c_5\xi_1\xi_2 + c_6\xi_1\xi_3 + c_7\xi_2\xi_3 + c_8\frac{1}{2}\left(3\xi_1^2-1\right) + c_9\frac{1}{2}\left(3\xi_2^2-1\right) + c_{10}\frac{1}{2}\left(3\xi_3^2-1\right).$$

Before we compute the Sobol indices we must estimate the total variance $\hat{V} = \sum_{k=2}^{10} c_k^2$. The First-order and Total-order Sobol indices of ξ_1, for example, can directly be computed as follows,

$$\mathcal{S}_1 = \frac{c_2^2 + c_8^2}{\hat{V}}, \quad \mathcal{S}_{T_1} = \frac{c_2^2 + c_5^2 + c_6^2 + c_8^2}{\hat{V}}.$$

3.1.3. Example: Mechanical Oscillator

In this section we shall illustrate the computation of Sobol indices using the mechanical oscillator example (1.21) from chapter 1. We recollect the model here for convenience,

$$\mathcal{M}(t,\alpha,\beta,\ell) = \ell e^{-\alpha t}(cos\beta t + \frac{\alpha}{\beta}sin\beta t).$$

The method introduced in section 3.1 is applicable for time-independent models $\mathcal{M}(\xi)$ with a scalar output. In order to demonstrate, we shall first make the above model time-independent by fixing $t = 2$s. This allows us to focus on the model $\mathcal{M}(2,\alpha,\beta,\ell)$ and compute the Sobol indices at that point.

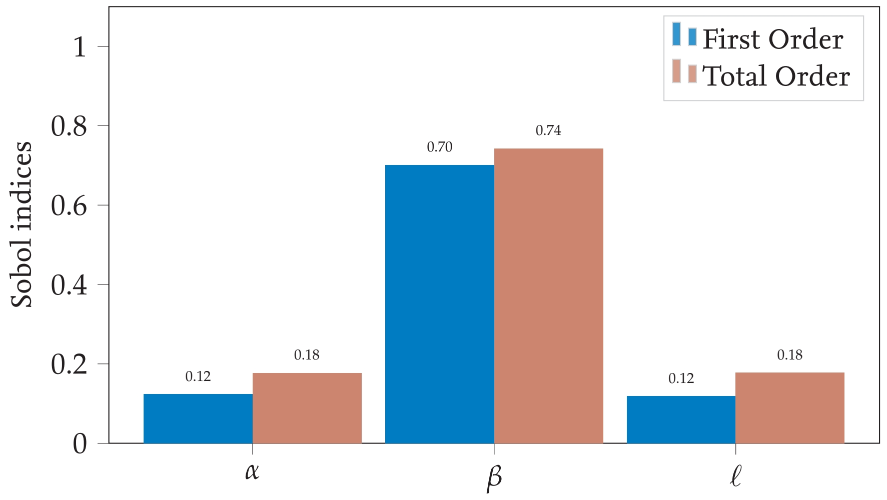


Figure 3.1.: First-order and Total-order Sobol indices of the mechanical oscillator example at $t = 2$s.

We use the Pick and Freeze approach to compute the Sobol indices using $N = 10^5$ Monte Carlo samples and plot the results in Figure 3.1. Although this leads to $N(N_p+2) = 10^5 * (3+2) = 5 \times 10^5$ models evaluations, it can be computed quickly since a closed form

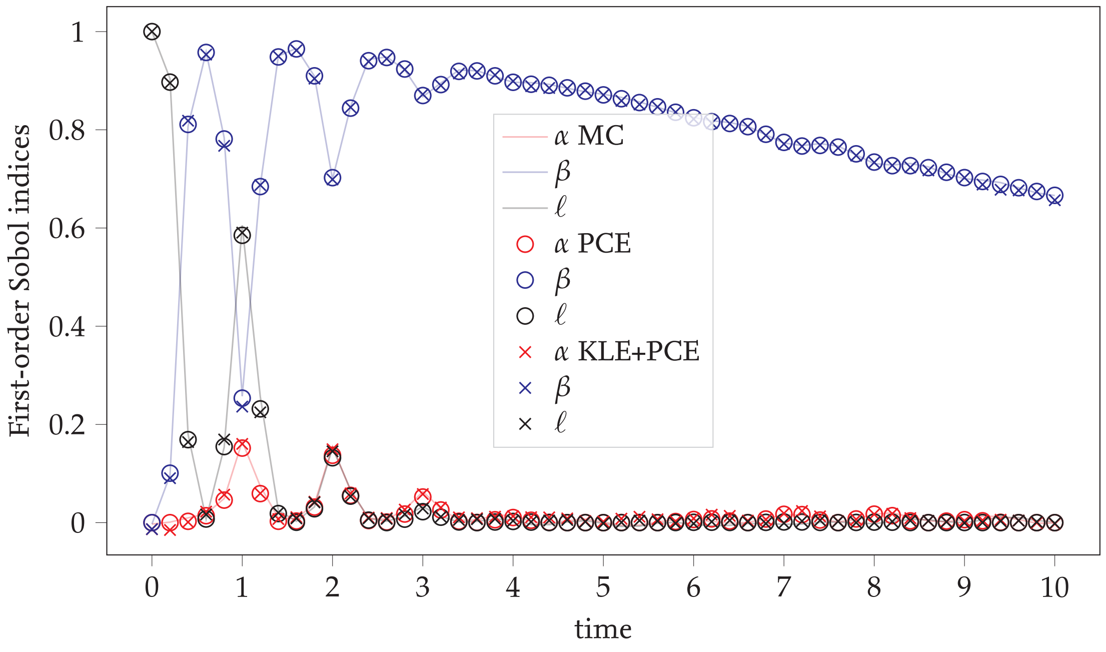


Figure 3.2.: First-order Sobol indices of the mechanical oscillator example computed using three approaches.

solution is available. In the plot we note that the variable β has the highest sensitivity and from the discussion in the section 3.1, we can conclude that this parameter has the largest contribution towards the output uncertainty and one must focus on reducing uncertainty in this parameter to reduce the output uncertainty at $t = 2$. Secondly, the differences between the First and Total-order indices is small, which implies that the interaction effects in the model at $t = 2$s are smaller compared to the First-order effects.

Using the same approach, we compute the Sobol indices at 51 equidistant points in $\mathcal{T} = [0, 10]$ and the model at each point is given by $\mathcal{M}(t_m, \alpha, \beta, \ell)$. The resulting First-order and Total-order Sobol indices are plotted in Figure 3.2 and 3.3 respectively (and the Total-order indices appear to be in agreement with the results in [3]). We shall refer to these indices as *Pointwise-in-time Sobol Indices*. The indices are computed using 3 approaches,

1. **MC**: The indices are computed using the Pick and Freeze Estimator introduced in 3.1.2. We use $N = 10^5$ Monte Carlo samples for each point in time to compute the indices and the evaluations necessary were computed directly using the model.

2. **PCE**: Time-frozen PCE surrogates are constructed at each of the 51 time instances and the Sobol indices are directly computed using the PCE coefficients. Each PCE surrogate has a total polynomial degree $p = 10$ and is trained using $N = 10^4$ MC samples. The trend in the LOO errors is similar to plot 2.3 and the maximum error is 0.014% at $t = 10$s.

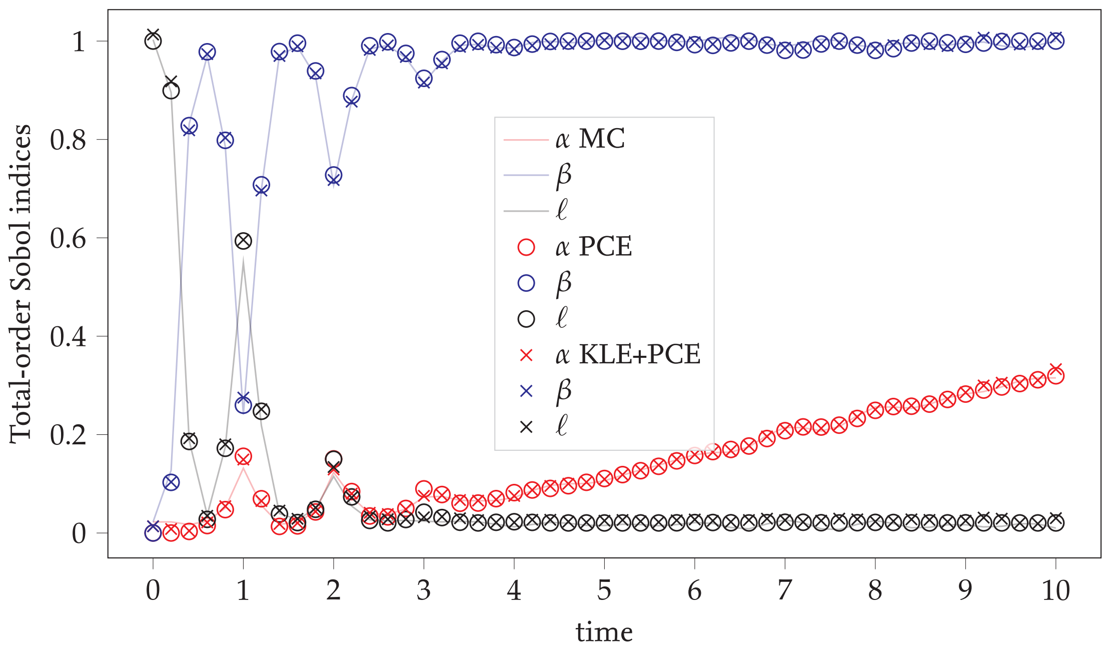


Figure 3.3.: Total-order Sobol indices of the mechanical oscillator example computed using three approaches.

3. **KLE+PCE**: A global surrogate using the KLE+PCE approach is created and the Sobol indices are computed using the Pick and Freeze approach. This is similar to the MC method above, except, the model evaluations are replaced by surrogate evaluations. To create the surrogate, the covariance matrix is approximated using $N = 10^4$ MC samples. Since the process is low-rank, the surrogate uses $N_{kl} = 10$ terms in the basis which explain 99.9995% of the variance. The resulting projections of the process on the basis are approximated using a gPCE with a total polynomial degree $p = 10$. The trend in the errors in similar to the plot in Figure 2.6 and the maximum LOO error is 0.09% in mode 10.

The Pointwise-in-time Sobol indices computed using the surrogate models seem to be in fair agreement with the MC Pick and Freeze approach. This indicates that the surrogate models could provide a reasonable estimate of the Sobol indices when the MC Pick and Freeze approach is intractable, given that the surrogate error is small.

However, the indices of the input variables change over time and making an inference about the sensitivities of the variables w.r.t to the entire process it not possible. This motivates us to consider a different index which we shall introduce in the following section.

3.2. Generalised Sobol indices

In the previous section we computed the pointwise-in-time Sobol indices for the mechanical oscillator example, which was a time-dependent process $\mathcal{M}(t, \boldsymbol{\xi})$. This approach is useful if we are interested in the sensitivities at a particular point in the deterministic domain, say t_m. Using the variance of the model at that point $\mathbb{V}[\mathcal{M}(t_m, \boldsymbol{\xi})]$ as our metric, we can compute the sensitivities of the stochastic inputs as illustrated in Figure 3.4.

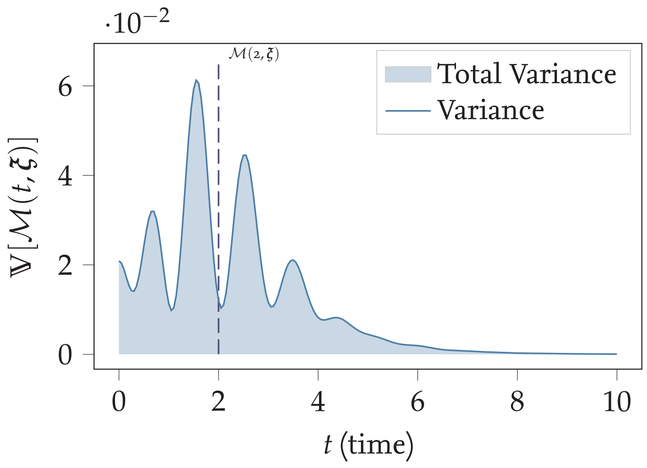


Figure 3.4.: Plot of Variance changing over time (blue curve) and the total variance of the process (shaded area) for the mechanical oscillator example.

However, Alexandrian et al. [3] point out that this metric is insufficient to provide sensitivities for the entire process and for processes with vector-valued outputs in general. Consider a scenario in the mechanical oscillator example, in which we are interested in reducing the uncertainty in the model output in the interval $\mathcal{T} = [0, 10]$. Since the sensitivities change over time, it is not possible to determine, holistically, which parameters have the largest contribution towards the overall uncertainty (refer Figure 3.2 and 3.3). To address this issue, Alexandrian et al. extend the idea from [15] and propose 'total variance of the process' as a new metric to compute sensitivities [1]. This is simply the the total area under the curve (shaded in blue) in Figure 3.4. This enables the measure to account for the correlation in the outputs, which the pointwise-in-time indices fail to capture. These new *history-aware* sensitivities are called *Generalised Sobol Indices* (GSI).

In the following section we shall provide a short overview of the mathematical framework to compute generalised Sobol Indices for models with vector-valued outputs and see that these indices are essentially the ratio of the integrals of the numerator and denominator to the Sobol indices. For further details refer [3, 15].

[1]One can extend this idea to random fields represented by $\mathcal{M}(Z, \boldsymbol{\xi}) : \mathcal{Z} \subseteq \mathbb{R}^q \times \Omega \rightarrow \mathbb{R}$, where $\mathcal{Z}$ could, for example, be a 2D spatial field $[-1, 1] \times [0, 1]$. Inline with the original nomenclature, one could call them *domain-aware* sensitivities. This has been implemented by the author for a 2D FEM problem, but for space reasons is not presented in this work. It can be found in the repository given in the preface.

3.2.1. Mathematical preliminaries

We begin by considering a centered, square integrable random process $Y = \mathcal{M}(t,\xi) : \mathcal{T} \times \mathbb{R}^{N_p} \to \mathbb{R}$ at m discretised points in the deterministic domain,

$$\mathcal{T}_m := \{t_1 < t_2 \ldots < t_m\} \in \mathcal{T},$$

to obtain a vector-valued model $\boldsymbol{Y} = \mathcal{M}_m(\xi) : \mathbb{R}^{N_p} \to \mathbb{R}^m$, where the j^{th} output corresponds to the random process evaluated at t_j,

$$(\mathcal{M}_m(\xi))_j := \mathcal{M}(t_j, \xi).$$

This perspective allows us to use the generalised Sobol Indices proposed for models with vector-valued outputs as in [15].

Similar to the Sobol Decomposition in (3.1) for scalar outputs, an HDMR for models with vector-valued outputs called the *Hoeffding Decomposition* can be given as follows [15],

$$\boldsymbol{Y} = \boldsymbol{Y}_u(\xi_u) + \boldsymbol{Y}_{u^{\complement}}(\xi_{u^{\complement}}) + \boldsymbol{Y}_{u,u^{\complement}}(\xi_u, \xi_{u^{\complement}}), \tag{3.19}$$

where $\boldsymbol{u}$ refers to a subset of $\{1, \ldots, N_p\}$ and $\boldsymbol{u}^{\complement}$ refers to its compliment w.r.t the same set. $\boldsymbol{Y}_u(\xi_u)$ and $\boldsymbol{Y}_{u^{\complement}}(\xi_{u^{\complement}})$ are functions of the random variables index by $\boldsymbol{u}$ and $\boldsymbol{u}^{\complement}$ respectively and the term $\boldsymbol{Y}_{u,u^{\complement}}(\xi_u, \xi_{u^{\complement}})$ captures their interaction. Similar to the Sobol decomposition, the terms in (3.19) are defined as follows:

$$\begin{aligned} \boldsymbol{Y}_u(\xi_u) &= \mathbb{E}[\boldsymbol{Y} \mid \xi_u]. \\ \boldsymbol{Y}_{u^{\complement}}(\xi_{u^{\complement}}) &= \mathbb{E}[\boldsymbol{Y} \mid \xi_{u^{\complement}}]. \\ \boldsymbol{Y}_{u,u^{\complement}}(\xi_u, \xi_{u^{\complement}}) &= \boldsymbol{Y} - \boldsymbol{Y}_u(\xi_u) - \boldsymbol{Y}_{u^{\complement}}(\xi_{u^{\complement}}). \end{aligned} \tag{3.20}$$

Following the idea of variance decomposition using HDMR in ANOVA for scalar outputs (3.5), it is possible to derive a covariance decomposition for the Hoeffding Decomposition,

$$\Sigma = \Sigma_u + \Sigma_{u^{\complement}} + \Sigma_{u,u^{\complement}}, \tag{3.21}$$

where $\Sigma, \Sigma_u, \Sigma_{u^{\complement}}, \Sigma_{u,u^{\complement}}$ denote the covariance of the terms $\boldsymbol{Y}, \boldsymbol{Y}_u(\xi_u), \boldsymbol{Y}_{u^{\complement}}(\xi_{u^{\complement}}), \boldsymbol{Y}_{u,u^{\complement}}(\xi_u, \xi_{u^{\complement}})$ in (3.19) respectively. For scalar outputs, this is equivalent to (3.5).

Unlike the scalar ANOVA decomposition, the terms in (3.21) are matrices and must be converted to scalars. To accomplish this, Gamboa et al. [15] chose the trace operator $\mathrm{Tr}(\cdot)$, because it is linear, satisfies invariance properties and it is easy to implement the Pick and Freeze estimator,

$$\mathrm{Tr}(\Sigma) = \mathrm{Tr}(\Sigma_u) + \mathrm{Tr}(\Sigma_{u^{\complement}}) + \mathrm{Tr}(\Sigma_{u,u^{\complement}}).$$

Finally we can define the First-order generalised Sobol Indices (3.22) and Total-order generalised Sobol Indices (3.23) of $\xi_u = \xi_i$ as follows:

$$\mathcal{G}_i(\mathcal{T}) := \frac{\mathrm{Tr}(\Sigma_{\xi_i})}{\mathrm{Tr}(\Sigma)}. \tag{3.22}$$

$$\mathcal{G}_{T_i}(\mathcal{T}) := \frac{\mathrm{Tr}(\Sigma_{\xi_i}) + \mathrm{Tr}(\Sigma_{\xi_i,\xi|\xi_i})}{\mathrm{Tr}(\Sigma)} = 1 - \frac{\mathrm{Tr}(\Sigma_{\xi|\xi_i})}{\mathrm{Tr}(\Sigma)}. \tag{3.23}$$

For an intuitive understanding of (3.22), let us consider the time discretised perspective of the random process mentioned above. The trace operator in the discretised case is simply the sum of the diagonals terms, which implies that the numerator is the sum of the variances of each of the terms in $\mathbb{E}[Y \mid \xi_1]$ and can be written as,

$$\mathrm{Tr}(\Sigma_{\xi_i}) = \sum_j \underbrace{\mathbb{V}_{\xi|\xi_i}\left[\ \mathbb{E}_{\xi|\xi_i}\left[\mathcal{M}(t_j,\xi) \mid \xi_i\right]\ \right]}_{=:\mathcal{S}_i^{num}(\mathcal{M};t_j)}. \tag{3.24}$$

The summand $\mathcal{S}_i^{num}(\mathcal{M};t_j)$ is the numerator of the First-order Sobol index of ξ_i for the pointwise-in-time model $\mathcal{M}(t_j,\boldsymbol{\xi})$ (3.22). Similarly the denominator is the trace of the covariance matrix of the random process,

$$\mathrm{Tr}(\Sigma) = \sum_j \underbrace{\mathbb{V}_{\xi}\left[\ \mathbb{E}_{\xi}\left[\mathcal{M}(t_j,\xi)\right]\right]}_{=:V(\mathcal{M};t_j)}. \tag{3.25}$$

The summand $V(\mathcal{M};t_j)$ is the variance of the pointwise-in-time model $\mathcal{M}(t_j,\boldsymbol{\xi})$. Since the trace is also the sum of eigenvalues, $\mathrm{Tr}(\Sigma)$ represents the total variance of the process. This implies that the GSI are the ratios of the sums of the numerator and sums of the denominator of the Sobol Indices. We allude once again to Figure 3.4 to illustrate the difference in the metrics used in the Sobol indices and GSI. The GSI use the total variance as a metric and Sobol indices use the variance at a specific point in time to compute sensitivities.

The details of these intuitions can be found in [3], where Alexandrian et al. prove the above for the functional case of a time-dependent process.

$$\mathcal{G}_i(\mathcal{T}) = \frac{\int_{\mathcal{T}} \mathcal{S}_i^{num}(\mathcal{M};t)\,\mathrm{d}t}{\int_{\mathcal{T}} V(\mathcal{M};t)\,\mathrm{d}t}. \tag{3.26}$$

$$\mathcal{G}_{T_i}(\mathcal{T}) = \frac{\int_{\mathcal{T}} \mathcal{S}_{T_i}^{num}(\mathcal{M};t)\,\mathrm{d}t}{\int_{\mathcal{T}} V(\mathcal{M};t)\,\mathrm{d}t}. \tag{3.27}$$

where $\mathcal{S}_{T_i}^{num}(\mathcal{M};t)$ is the numerator of the Total-order Sobol index of ξ_i of $\mathcal{M}(t,\boldsymbol{\xi})$ (3.23).

3.2.2. Practical Computation

For most practical problems, the response of the model $\mathcal{M}(t,\xi)$ is available at a finite number of points in time. In order to compute the GSI for $\xi_i, i = \{1,\dots,N_p\}$, we consider approximations of the equations (3.22) and (3.23),

$$\mathcal{G}_i(\mathcal{T}) \approx \frac{\sum_{j=1}^{N_\text{quad}} w_j\, \mathcal{S}_i^{num}(\mathcal{M};t_j)}{\sum_{j=1}^{N_\text{quad}} w_j\, V(\mathcal{M};t_j)}, \quad \mathcal{G}_{T_i}(\mathcal{T}) \approx \frac{\sum_{j=1}^{N_\text{quad}} w_j\, \mathcal{S}_{T_i}^{num}(\mathcal{M};t_j)}{\sum_{j=1}^{N_\text{quad}} w_j\, V(\mathcal{M};t_j)}, \tag{3.28}$$

where the integrals are approximated using quadrature weights $\{w_j\}_{j=1}^{N_\text{quad}}$ and model evaluations at quadrature nodes $\{t_j\}_{j=1}^{N_\text{quad}} \in \mathcal{T}$.

Using Pick and Freeze estimator

Computation of the Sobol Indices using the Pick and Freeze estimator requires $N(N_p+2)$ evaluations of the model $\mathcal{M}(\xi) \in \mathbb{R}$ and 3 matrices of size $\mathbb{R}^N$ containing the model evaluations must be stored. However, computing the GSI using the Pick and Freeze Estimator, can either have higher memory requirements or require larger number of model evaluations.

Assuming evaluations of the model $\mathcal{M}(t,\xi)$ provides the entire time response $\{t_1 < t_2 \dots < t_{N_\text{quad}}\} \in \mathcal{T}$, the number of model evaluations required would not change. However, memory requirements would significantly increase. We must now store model evaluations $\mathcal{Y}_\mathcal{A}, \mathcal{Y}_\mathcal{B}, \mathcal{Y}_{\mathcal{C}_i} \in \mathbb{R}^{N_\text{quad}\times N}, i \in \{1,\dots,N_p\}$ corresponding to the input matrices $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, \boldsymbol{\mathcal{C}}_i \in \mathbb{R}^{N\times N_p}, i \in \{1,\dots,N_p\}$ as previously defined. The computations of the indices using equations (3.2) and (3.4) are negligible compared to the model evaluations.

In case one model evaluation provides the response for a single t_i, the computation of the GSI will require $N(N_p+2)N_\text{quad}$ model evaluations.

Using gPCE surrogates

To circumvent the high computational cost of the Pick and Freeze estimators we shall consider the gPCE surrogate $\hat{\mathcal{M}}^{PC}(\xi)$ from section 2.1 as in [3]. This is an efficient method to compute the GSI indices because the PCE approximations allow direct computation of indices as discussed this section 3.1.2. Use of the surrogate allows us to rewrite (3.28) as follows:

$$\mathcal{G}(\mathcal{T}) \approx \frac{\sum_{j=1}^{N_\text{quad}} w_j\, \mathcal{S}^{num}(\hat{\mathcal{M}}^{PC};t_j)}{\sum_{j=1}^{N_\text{quad}} w_j\, V(\hat{\mathcal{M}}^{PC};t_j)}, \quad i = \{1,\dots,N_p\}, \tag{3.29}$$

where the subscripts i and T_i corresponding to the First-order and Total-order GSI respectively are dropped for brevity.

We substitute the truncated gPCE approximation from (2.18) in (3.29) to obtain,

$$\mathcal{G}(\mathcal{T}) \approx \frac{\sum_{j=1}^{N_{\text{quad}}} w_j\, \mathcal{S}^{num}\left(\sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}); t_j\right)}{\sum_{j=1}^{N_{\text{quad}}} w_j\, V\left(\sum_{k=1}^{N_{PC}} c_k \Psi_k(\boldsymbol{\xi}); t_j\right)}. \tag{3.30}$$

The use of the gPCE surrogate allows us to compute the sensitivity coefficients directly and can be summarised as follows,

GSI with gPCE surrogates

$$\mathcal{G}(\mathcal{T}) \approx \left(\sum_{j=1}^{N_{\text{quad}}} \sum_{k\in\mathcal{K}} w_j\, c_k^2(t_j)\right) \Bigg/ \left(\sum_{j=1}^{N_{\text{quad}}} \sum_{k=2}^{N_{PC}} w_j\, c_k^2(t_j)\right). \tag{3.31}$$

In order to obtain First-order generalised Sobol indices, we use the First-order coefficients picker $\mathcal{K} = \mathcal{K}_i$ (3.17) and similarly $\mathcal{K} = \mathcal{K}_{T_i}$ for the Total-order generalised Sobol indices (3.18). At first, the terms in the coefficient pickers may seem daunting, but they are doing a very simple task. The First-order picker essentially picks all the coefficients of the univariate polynomials of ξ_i and the Total-order picker picks all the coefficients that contain the term ξ_i.

In this work we use a Gaussian quadrature to approximate the integrals, which allows us to estimate the integral with a small number of function evaluations. However, there is one small drawback. The model response at these quadrature points must be interpolated since usually a uniformly discretised response is available. But this is somewhat counteracted by the uniform high-fidelity response available for a good interpolation. (Overall the results seem to be in agreement with [3] for mechanical oscillator example). Alternatively, one can use the trapezoidal quadrature to avoid interpolation.

Using KLE+PCE surrogate

Although the GSI can be obtained by constructing pointwise-in-time PCE surrogates, the surrogate approximation is known to degrade over time (however, the rate and amount of degradation is problem specific [20]). KLE+PCE surrogates are a good alternative for low-rank processes since they represent the model in the eigenbasis of the covariance operator and thus exploits the problem structure. We begin by recollecting the KLE+PCE approximation for a centred random process in (2.50),

$$\mathcal{M}(t,\boldsymbol{\xi}) = \sum_{j=1}^{\infty} \Phi_j(\boldsymbol{\xi}) \boldsymbol{e}_j(t) = \sum_{j=1}^{\infty}\sum_{k=1}^{\infty} f_{\Phi_j}^{(k)} \mathcal{F}_{\Phi_j}^{(k)}(\boldsymbol{\xi}) \boldsymbol{e}_j(t). \tag{3.32}$$

As a reminder, the KLE+PCE expansion uses PCE to approximate the projections $\Phi_j(\boldsymbol{\xi})$ of the random process on the KLE basis. Using the expansion in the above equation, we can compute the numerator of the First-order generalised Sobol Indices in (3.26),

$$\int_{\mathcal{T}} \mathcal{S}_i^{num}(\mathcal{M}^{KL+PCE};t)\,\mathrm{d}t = \int_{\mathcal{T}} \mathcal{S}_i^{num}\left(\sum_{j=1}^{\infty}\sum_{k=1}^{\infty} f_{\Phi_j}^{(k)}\mathcal{F}_{\Phi_j}^{(k)}(\boldsymbol{\xi})v_j(t);t\right)\,\mathrm{d}t, \tag{3.33}$$

$$= \sum_{j=1}^{\infty}\sum_{k\in\mathcal{K}_i}\left(f_{\Phi_j}^{(k)}\right)^2. \tag{3.34}$$

The second equality in the above expression is proven in [3]. The term $\mathcal{K}_i$ represents the First-order coefficient picker (3.17) as explained in the gPCE surrogate. Therefore the numerator of the First-order GSI can be computed by squaring the univariate terms ξ_i in the gPCE projection across all basis terms. Similarly we can use the Total-order coefficients picker (3.18) for the numerator of the Total-order GSI. The denominator of the index is essentially the total variance of the process, given by the sum of the eigenvalues,

$$\int_{\mathcal{T}} V\left(\hat{\mathcal{M}}^{KL+PC}(t,\boldsymbol{\xi});t\right)\,\mathrm{d}t = \int_{\mathcal{T}} V\left(\sum_{j=1}^{\infty}\Phi_j(\boldsymbol{\xi})v_j(t);t\right)\,\mathrm{d}t, \tag{3.35}$$

$$= \sum_{i=j}^{\infty}\lambda_j. \tag{3.36}$$

For practical computations, we consider the truncated version of (3.32),

$$\hat{\mathcal{M}}^{KL+PC}(t,\boldsymbol{\xi}) \approx \sum_{j=1}^{N_{kl}}\Phi_j(\boldsymbol{\xi})v_j(t) \approx \sum_{j=1}^{N_{kl}}\sum_{k=1}^{N_{PC}} f_{\Phi_j}^{(k)}\mathcal{F}_{\Phi_j}^{(k)}(\boldsymbol{\xi})v_j(t), \tag{3.37}$$

where N_{kl} is the number of terms in the eigenbasis and N_{PC} represents the numbers of terms in the gPCE expansion of the projection. The computation of the GSI using the truncated KLE+PCE basis can be summarised as follows:

GSI with KLE+PCE surrogates

$$\mathcal{G}(\mathcal{T}) \approx \frac{\sum_{j=1}^{N_{kl}}\sum_{k\in\mathcal{K}}\left(f_{\Phi_j}^{(k)}\right)^2}{\sum_{j=1}^{N_{kl}}\lambda_j}, \tag{3.38}$$

where $\mathcal{K}$ is a coefficient picker.

3.2.3. Example: Mechanical Oscillator

In this section we compute the generalised Sobol indices for the mechanical oscillator example using the algorithms introduced in the previous section.

For computing the GSI using the Pick and Freeze Estimator, we use the relation (3.28) with $N = 10^6$ and a Gauss-Legendre quadrature with $N_{quad} = 30$ points. The plot of the evolution of the Total-order generalised Sobol index using this approach is plotted in Figure 3.5. Here, the index $\mathcal{G}_{T_i}([0,t_i])$ is computed at $m = 100$ equidistant points $t_i \in \mathcal{T} = [0.1, 10]$ with $i = \{1, \ldots, m\}$. The plot clearly indicates the dominance of the parameter β in governing the output uncertainty of the random process. Unlike the pointwise-in-time approach, the GSI are a holistic and consistent sensitivity metric for time-dependent processes. However, the indices of α and β are negative which indicates that despite using a large number of MC samples, the indices have not converged.

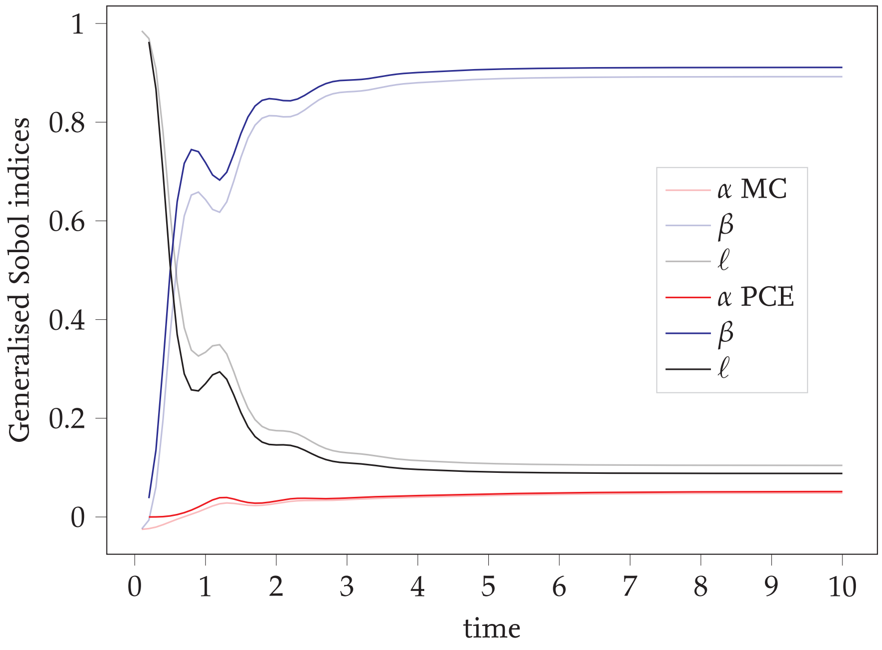


Figure 3.5.: Evolution of the generalised Total-order Sobol indices of the mechanical oscillator example computed using two approaches.

Next, we compute the Total-order GSI using the PCE approach summarised in (3.2.2). For computing the time-frozen PCE surrogates at the quadrature points, we employ a gPCE of degree $p = 10$ and train it using $N = 10^4$ MC samples. As in the previous case, we compute the evolution of the Total-order GSI over time using a Gauss-Legendre quadrature with $N_{quad} = 25$ points and plot it in the same Figure to illustrate the difference. The difference in the values could be attributed to the slow convergence of the MC estimator. Overall, the curves obtained using the PCE surrogate seem to be in fair agreement with the results in [3].

Finally, we compute the GSI using the KLE+PCE surrogate as summarised in (3.2.2). The surrogate is trained using $N = 10^4$ MC samples and uses a truncation order $N_{kl} = 10$. The projections of the random process on the basis are approximated using a gPCE of degree $p = 10$. A comparison of the Total-order GSI $\mathcal{G}(\mathcal{T})$ is shown in Figure 3.6. The results from the three approaches seem to be consistent and are in good agreement with the results in [3] and thus validates the implementation.

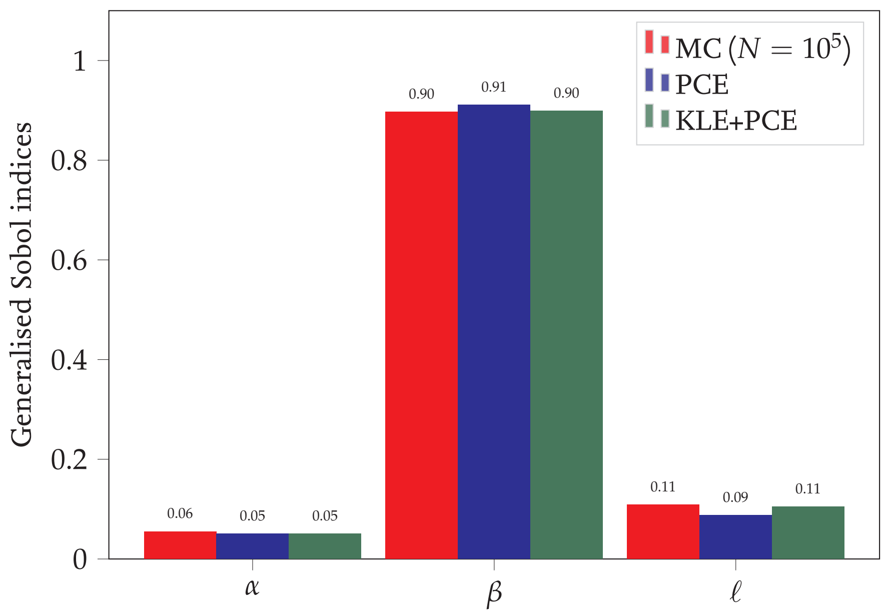
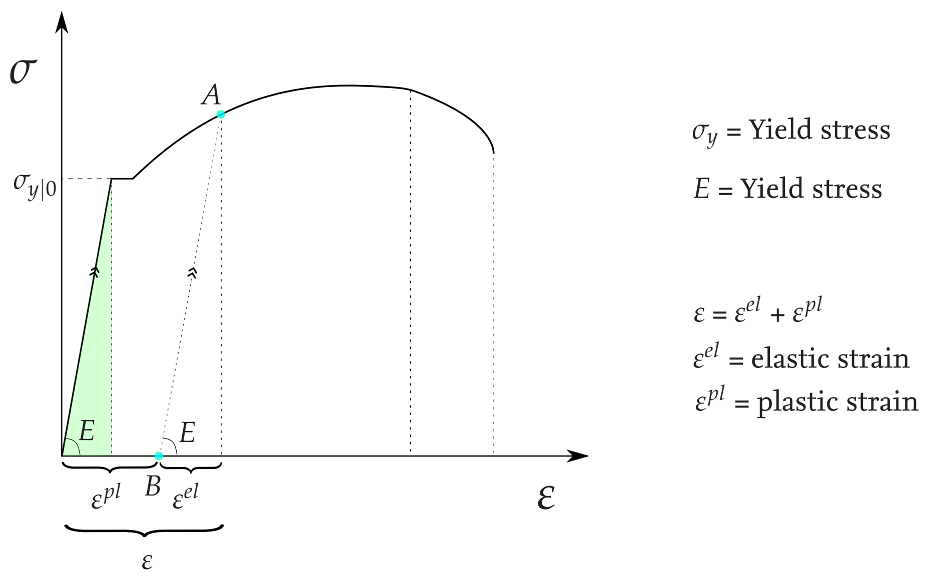


Figure 3.6.: Evolution of the generalised total-order Sobol indices of the mechanical oscillator example computed using three approaches.

4. Application Example: Chaboche Model

The Chaboche Model is a constitutive material model that describes the behaviour of viscoplastic materials such as steel. The basic model allows us to model the viscoplastic behaviour of materials but it can be extended to account for more advanced phenomenology such as temperature effects, strain memorization etc. [12]. However in this work, we shall restrict our focus to the basic model to study the sensitivities of the model parameters.

4.1. Elastoplasticity in 1D



σ
A
$\sigma_{y|0}$
E
E
ε^{pl}
B
ε^{el}
ε
ε
σ_y = Yield stress
E = Yield stress
$\varepsilon = \varepsilon^{el} + \varepsilon^{pl}$
ε^{el} = elastic strain
ε^{pl} = plastic strain

Figure 4.1.: Strain-strain curve from a 1D tensile test of a steel bar.

To introduce the idea of the Chaboche model, we shall first consider the simple 1D tensile test of a steel bar. In a typical experiment, force F is applied to the specimen and the resulting strain ε in the specimen is measured (or alternatively, a strain ε could applied on the specimen and the force F required to apply the strain could be measured). In the 1D experiment, the stress is simply, $\sigma = \frac{F}{A}$, where A is the cross-section area of the material.

A typical stress-strain curve for metals is illustrated in Figure 4.1. The area shaded in green denotes the elastic region of the material, where, the bar reverts back to the its original length upon unloading. In this region, the stress is directly proportional to the strain and the constant of proportionality is called the Young's Modulus E. This is called the *Hooke's law*. The *yield limit* σ_y marks the end of the elastic region ($\sigma_{y|0}$ in Figure 4.1) and the same holds true in compression [10].

Upon loading the specimen beyond yield limit the material does not fully revert back to its initial configuration and undergoes some permanent deformation. For example, if the bar is loaded up to some point A (in Figure 4.1) in the plastic region, upon unloading, the material follows a straight line parallel to the elastic region with slope E and reaches a point B after the external strain is fully removed. The part of the strain ε that we regain is aptly termed the elastic strain ε^{el} and the permanent strain is called the plastic strain ε^{pl}. From this it follows:

$$\varepsilon = \varepsilon^{el} + \varepsilon^{pl}. \tag{4.1}$$

The above equation is called the additive decomposition of strains. It must be noted that the strains considered here are infinitesimally small. From the constitutive relation, the stress at a point is only caused by the elastic strains,

$$\sigma = E\varepsilon^{el} = E(\varepsilon - \varepsilon^{pl}). \tag{4.2}$$

Upon reloading the deformed specimen, we note that the yield limit of the material is no longer $\sigma_{y|0}$ but has increased. The increase in the yield limit of the material due to plastic strains is termed as *hardening*. From our everyday experience we know that it requires more effort to straighten a bent metal spoon than to bend it. Upon bending, the spoon undergoes hardening which increases its yield limit and therefore requires more effort to bring it back its original configuration.

After the initial yield limit is exceeded, material behaviour can be classified into 2 types: *ideal plasticity* and *viscoplasticity*. Ideal plasticity refers to the rate-independent deformation of the material, which implies that the stress in the material remains constant even when the strain in the material is increased. On the other hand, the rate-dependent behaviour is called viscoplasticity in which the stress state of the material depends on the strain rate. The two concepts are illustrated in Figure 4.2 for different strain rates $\dot{\varepsilon}$.

This motivates us to model the development of stress dependent on the strain rate. First we rewrite (4.1) in the rate form in equation (4.3) and use equation (4.2) to obtain (4.4).

$$\dot{\varepsilon} = \dot{\varepsilon}^{el} + \dot{\varepsilon}^{pl}. \tag{4.3}$$

$$\dot{\sigma} = E(\dot{\varepsilon} - \dot{\varepsilon}^{pl}). \tag{4.4}$$

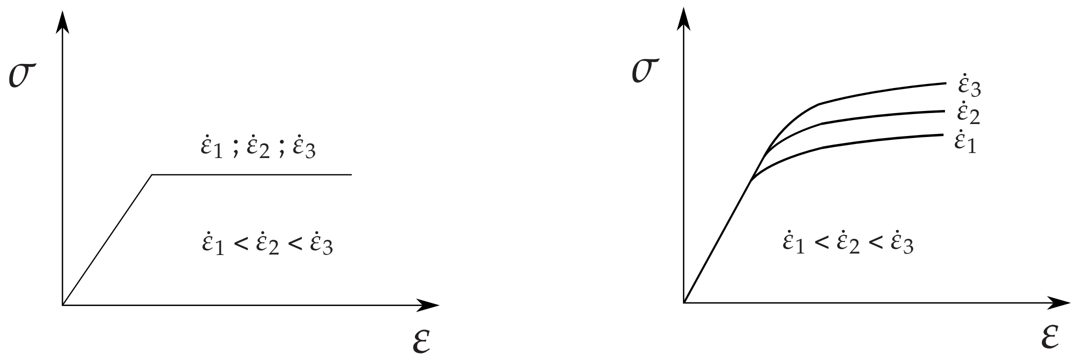


Figure 4.2.: Ideal plasticity (left) and viscoplasticity (right) with different strain-rates. [18]

4.2. Yield Function, Flow Rule and Hardening

From the previous section it is now clear that plastic strains cause hardening in the material. It is therefore intuitive to decompose this phenomenon in three parts [10]:

- Yield criterion: A function that assess if the material is undergoing yielding.
- Flow Rule: A function to determine the evolution of the plastic strains.
- Hardening: A relation between the evolution of the yield limit σ_y w.r.t. the plastic strains.

Yield Function

We define the yield function f as follows:

$$f(\sigma, \sigma_y) = |\sigma| - \sigma_y \leq 0, \tag{4.5}$$

where $f < 0$ means the material is in the elastic region, $f = 0$ implies yielding and $f > 0$ is not admissible. Before the material undergoes any plastic strains, $\sigma_y = \sigma_{y|0}$, where $\sigma_{y|0}$ denotes the initial yield limit.

Flow Rule

The plastic strain rate is given as,

$$\dot{\varepsilon}^{pl} = \begin{cases} +\dot{\lambda}, & \text{if } \sigma > 0, \\ -\dot{\lambda}, & \text{if } \sigma < 0. \end{cases} \tag{4.6}$$

Using $\frac{df}{d\sigma} = sign(\sigma)$ from equation (4.5), we can rewrite equation (4.6) as,

$$\dot{\varepsilon}^{pl} = \dot{\lambda}\frac{df}{d\sigma}. \tag{4.7}$$

Equation (4.7) is called the flow rule. The flow rule has two terms, $\dot{\lambda}$ called the plastic multiplier and $\frac{df}{d\sigma}$ which determines the direction, either tension or compression.

We now rewrite the two rules (4.5) and (4.6) from above,

$$\dot{\lambda} \geq 0 \quad \text{and} \quad f(\sigma, \sigma_y) \leq 0. \tag{4.8}$$

Since $\dot{\lambda} = 0$ and $f(\sigma, \sigma_y) < 0$ when the bar is deforming elastically and $\dot{\lambda} > 0$ and $f(\sigma, \sigma_y) = 0$ during plastic deformation. Therefore we can combine equation (4.8) as,

$$\dot{\lambda} f(\sigma, \sigma_y) = 0. \tag{4.9}$$

This is called the *Karush-Kuhn-Tucker (KKT) condition*. Upon taking the derivative of equation (4.9), we obtain the *consistency condition*:

$$\dot{\lambda}\dot{f}(\sigma, \sigma_y) = 0 \qquad \Longrightarrow \dot{f}(\sigma, \sigma_y) = 0. \tag{4.10}$$

Hardening

The evolution of the yielding limit is governed by the plastic strains. A simple example of evolution of yield limit σ_y with plastic strains is the linear law,

$$\dot{\sigma}_y = H\dot{\lambda}.$$

where H is the plastic modulus. The yield limit of the material changes based on H, where $H > 0$ is called hardening, $H = 0$ is perfect plasticity and $H < 0$ is called softening. It can be shown that for that linear case above, $H \in [-E, E]$ [p.19 10], which means that the change in the yield limit is bounded from above and below.

4.3. Isotropic and Kinematic Hardening

There are primarily two ways in which the yield limit changes, *isotropic hardening* and *kinematic hardening*.

Isotropic Hardening: The yield limit changes by $\Delta\sigma_y$ in tension and compression, which causes the elastic region to grow uniformly.

$$f(\sigma, \sigma_{y|0}, K) = |\sigma| - \underbrace{(\sigma_{y|0} + K)}_{\sigma_y} \leq 0. \tag{4.11}$$

In equation (4.11), $K = \Delta\sigma_y$ refers to the isotropic hardening.

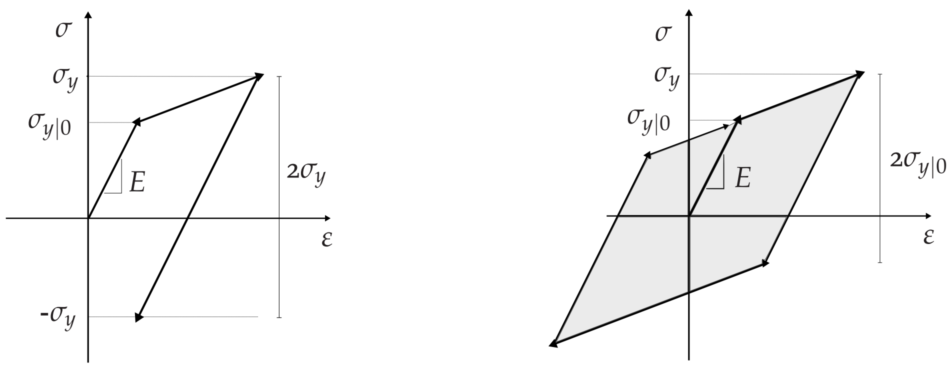


Figure 4.3.: (Left) Isotropic Hardening and (Right) Kinematic Hardening [10]

Kinematic Hardening: The yield limit changes by $\Delta\sigma_y$ in tension and changes by $-\Delta\sigma_y$ in compression, which only causes a translation of the elastic region. The asymmetric change in the yield limit in tension and compression is called the *Bauschinger effect*. The idea of translation of the yield surface is modelled using the notion of *back stress* [10].

$$f(\sigma, \sigma_{y|0}, X) = |\sigma - X| - \sigma_{y|0} \leq 0. \tag{4.12}$$

In equation (4.12), X refers to the back stress and is the amount by which the center of the elastic region translates and we shall call the term $|\sigma - X|$ *effective stress* and denote it by σ_v.

The effects of isotropic and kinematic hardening are illustrated in Figure 4.3. In case of isotropic hardening, the yield limit increases from $\sigma_{y|0}$ to σ_y in both tension and compression which causes the elastic region to expand. However for kinematic hardening, the elastic region remains constant (depicted by the shaded area) and an increase in the yield limit in tension causes a decrease in yield limit in compression.

Upon combining the isotropic and kinematic hardening, we obtain a yield function as follows:

$$f(\sigma, \sigma_{y|0}, K, X) = |\sigma - X| - \underbrace{(\sigma_{y|0} + K)}_{\sigma_y} \leq 0. \tag{4.13}$$

Previously, we enforced that the yield function be negative semi-definite $f \leq 0$. However now in the context of the Chaboche Model, $f > 0$ is permitted. This also implies that the consistency condition (equation (4.10)) is no longer valid. The stresses exceeding the elastic region, called *excess stress* denoted by σ_{ex}, are now used to determine the plastic strains in the material [18].

$$\sigma_{ex} := \sigma_v - \sigma_y. \tag{4.14}$$

The plastic multiplier $\dot{\lambda}$ is defined in equation (4.15), where the terms D and n are material parameters and must be determined experimentally and the term $\dot{\varepsilon}_0[1/s]$ balances the units on both sides. The Macaulay brackets $\langle\cdot\rangle$ are used to ensure that the effects are modelled only when the excess stress is positive, i.e the stress state lies outside the elastic region [18].

$$\dot{\lambda} := \left\langle \frac{\sigma_{ex}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0 \qquad \langle x \rangle = \begin{cases} 0 & \text{if} \quad x \leq 0 \\ x & \text{if} \quad x > 0 \end{cases} \tag{4.15}$$

4.4. ODE system

The equations modeling the evolution of the isotropic hardening and back stresses are derived from the first principles of irreversible processes in thermodynamics. We shall only use the final equations in this work [12, 4, 39]. Interested readers may refer to [17] for a detailed derivation of the equations. For completeness, the equations for the 3D model are presented first and will later be simplified to the 1D case to continue the discussion from the previous section. The system of equations contains 3 quantities, strain rate $\dot{\varepsilon} \in \mathbb{R}^6$, evolution of isotropic hardening $\dot{K} \in \mathbb{R}$ and evolution of back stresses $\dot{\mathbf{X}} \in \mathbb{R}^6$.

Before we proceed to describing the 3D Chaboche model, we shall briefly recollect the 3D constitutive equations and the notion of *equivalent stress.* The rate dependent constitutive relation in 3D is given as,

$$\underbrace{\begin{bmatrix} \dot{\varepsilon}_{xx} \\ \dot{\varepsilon}_{yy} \\ \dot{\varepsilon}_{zz} \\ \dot{\gamma}_{xy} \\ \dot{\gamma}_{xz} \\ \dot{\gamma}_{yz} \end{bmatrix}}_{\dot{\varepsilon}^{el}} = \underbrace{\frac{1}{E} \begin{bmatrix} 1 & -\nu & -\nu & 0 & 0 & 0 \\ -\nu & 1 & -\nu & 0 & 0 & 0 \\ -\nu & -\nu & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2(1+\nu) & 0 & 0 \\ 0 & 0 & 0 & 0 & 2(1+\nu) & 0 \\ 0 & 0 & 0 & 0 & 0 & 2(1+\nu) \end{bmatrix}}_{E^{-1}} \underbrace{\begin{bmatrix} \dot{\sigma}_{xx} \\ \dot{\sigma}_{yy} \\ \dot{\sigma}_{zz} \\ \dot{\tau}_{xy} \\ \dot{\tau}_{xz} \\ \dot{\tau}_{yz} \end{bmatrix}}_{\dot{\sigma}} . \tag{4.16}$$

where, ν is the poisson's ratio (typically 0.3 for steel). The notion of yielding in 1D is carried forward to 3D using the idea of *equivalent stress*, which captures the essence of yielding and allows us to define a yield function in higher dimensions and under complex loading scenarios. To define the equivalent stress, we first consider the 3D stress tensor,

$$\begin{bmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{bmatrix} .$$

The stress tensor can be split in two parts, hydrostatic stress S^H, which changes the volume of the material and deviatoric stress S^D, which changes the shape of the material.

$$I_1 = \frac{1}{3}\left(\sigma_{xx} + \sigma_{yy} + \sigma_{zz}\right).$$

$$\begin{bmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{bmatrix} = \underbrace{\begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_1 & 0 \\ 0 & 0 & I_1 \end{bmatrix}}_{S^H} + \underbrace{\begin{bmatrix} \sigma_{xx} - I_1 & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} - I_1 & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} - I_1 \end{bmatrix}}_{S^D}. \tag{4.17}$$

According to the von-Mises yield criterion, the material yields when the deviatoric stress J_2^D reaches a critical value, where J_2^D is the second invariant of the deviatoric stress tensor S^D.

$$\begin{aligned} J_2^D &= \frac{1}{2}\left(s_x^2 + s_y^2 + s_z^2\right) + \tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2, \\ s_x &= \sigma_{xx} - I_1, \\ s_y &= \sigma_{yy} - I_1, \\ s_z &= \sigma_{zz} - I_1. \end{aligned} \tag{4.18}$$

The final expression for equivalent stress in 3D is given as,

$$\sigma_v = \sqrt{\frac{1}{2}\left[\left(\sigma_{xx} - \sigma_{yy}\right)^2 + \left(\sigma_{yy} - \sigma_{zz}\right)^2 + \left(\sigma_{zz} - \sigma_{xx}\right)^2\right] + 3\left(\sigma_{xy}^2 + \sigma_{yz}^2 + \sigma_{zx}^2\right)}. \tag{4.19}$$

Evolution of strains

The evolution of the strains given in equation (4.20) can be obtained by combining equations (4.4) and (4.6), where $\boldsymbol{E}^{-1}$ is the Hooke's law in 3D and the vectors $\boldsymbol{\varepsilon} \in \mathbb{R}^6$ and $\boldsymbol{\sigma} \in \mathbb{R}^6$ contain the unique terms from the strain and stress tensor respectively [12].

$$\dot{\boldsymbol{\varepsilon}} = \dot{\boldsymbol{\varepsilon}}_{el} + \dot{\boldsymbol{\varepsilon}}_{pl} = \boldsymbol{E}^{-1} \cdot \dot{\boldsymbol{\sigma}} + \frac{\partial \sigma_{ex}}{\partial \boldsymbol{\sigma}} \left\langle \frac{\sigma_{ex}}{D} \right\rangle^n. \tag{4.20}$$

Evolution of isotropic hardening

In the Chaboche model, the isotropic hardening is modelled using the *Voce law* (equation (4.21)) which an exponential form of hardening [39]. This allows us to model the isotropic hardening more realistically as compared to the linear law in which the isotropic hardening approached infinity as the plastic strains reach infinity, which is not true in reality.

$$\dot{K} = b_{iso} \cdot \left(Q_{iso} - K\right) \cdot \left\langle \frac{\sigma_{ex}}{D} \right\rangle^n. \tag{4.21}$$

In case of the exponential law, the isotropic hardening K is initially 0 (before the material undergoes any plastic strains) and reaches an asymptotic/saturation value of Q_{iso} as plastic strains reach infinity and the parameter b_{iso} controls the rate at which the saturation value is reached. To demonstrate this more clearly, we rewrite the above equation as $K = Q_{iso}\left(1 - e^{-b_{iso}\varepsilon^{pl}}\right)$ [39].

Evolution of kinematic hardening

The evolution of back stresses is also governed by the plastic strains and is given in equation (4.22). In the 3D case, the back stresses $\boldsymbol{X} \in \mathbb{R}^6$ describes the evolution of the position of the center of the elastic region. Similar to the isotropic hardening, the parameter $\frac{2}{3}Q_{kin}$ is the saturation value of the back stress and the b_{kin} controls the rate at which the saturation value is reached.

$$\dot{\boldsymbol{X}} = b_{kin}\Big(\frac{2}{3}Q_{kin} \cdot \frac{\partial\sigma_v}{\partial\boldsymbol{\sigma}} - \boldsymbol{X}\Big) \cdot \Big\langle \frac{\sigma_{ex}}{D} \Big\rangle^n . \tag{4.22}$$

It is important to note that the back stresses or the Bauschinger effect, is only observed under cyclic loading, i.e. when the strains/strains reach the yield limits in both tension and compression. For monotonic loading, this cannot be observed [10].

1D Chaboche Model

We now simplify the $3D$ equations above to demonstrate the phenomenon in $1D$. For the $1D$ case, evolution of strains ((4.20)) reduces to

$$\dot{\varepsilon} := \dot{\varepsilon}_{el} + \dot{\varepsilon}_{pl} = E^{-1} \cdot \dot{\sigma}_{xx} + \frac{\partial\sigma_{ex}}{\partial\sigma_{xx}} \cdot \Big\langle \frac{\sigma_{ex}}{D} \Big\rangle^n \cdot \dot{\varepsilon}_0, \tag{4.23}$$

where, σ_{xx} makes the one dimensional simplification explicitly clear. We can use $\sigma_v = |\sigma_{xx} - X_{xx}| = \sqrt{(\sigma_{xx} - X_{xx})^2}$ and $\frac{\partial\sigma_{ex}}{\partial\sigma_{xx}} = \frac{\partial\sigma_v}{\partial\sigma_{xx}}$ (equation (4.14)) to expand the above expression,

$$\dot{\varepsilon} = E^{-1} \cdot \dot{\sigma}_{xx} + \frac{1}{\sigma_v}(\sigma_{xx} - X_{xx}) \cdot \Big\langle \frac{\sigma_{ex}}{D} \Big\rangle^n \cdot \dot{\varepsilon}_0. \tag{4.24}$$

Similarly we can simplify the back stress evolution ODE as,

$$\dot{X}_{xx} = b_{kin}\Big(\frac{2}{3}Q_{kin} \cdot \frac{1}{\sigma_v}(\sigma_{xx} - X_{xx}) - X_{xx}\Big) \cdot \Big\langle \frac{\sigma_{ex}}{D} \Big\rangle^n . \tag{4.25}$$

The ODE for the isotropic hardening remains unchanged and therefore the system of equations for $1D$ Chaboche Model can be summarised as,

$$\begin{bmatrix} \dot{\varepsilon}_{xx} \\ \dot{K} \\ \dot{X}_{xx} \end{bmatrix} = \begin{bmatrix} E^{-1} \cdot \dot{\sigma}_{xx} + \frac{1}{\sigma_v}(\sigma_{xx} - X_{xx}) \cdot \langle \frac{\sigma_{ex}}{D} \rangle^n \cdot \dot{\varepsilon}_0 \\ b_{iso} \cdot (Q_{iso} - K) \cdot \langle \frac{\sigma_{ex}}{D} \rangle^n \\ b_{kin}\Big(\frac{2}{3}Q_{kin} \cdot \frac{1}{\sigma_v}(\sigma_{xx} - X_{xx}) - X_{xx}\Big) \cdot \langle \frac{\sigma_{ex}}{D} \rangle^n \end{bmatrix}. \tag{4.26}$$

The model has 8 parameters, namely, $E, \sigma_{y|0}, Q_{iso}, b_{iso}, n, D, Q_{kin}, b_{kin}$ that must be calibrated.

4.5. Time Integration

To solve the resulting system of ODEs (4.26), we employ a numerical time integration scheme as it is not feasible to solve the system of non-linear ODEs analytically. There are many time integration techniques available to solve ODEs [30, p.310], however we shall only present two one-step methods, Euler Forward and Euler Backward.

For convenience, we briefly recollect the two schemes before the implementation is illustrated. Consider a first-order ODE of the form [30]:

$$\dot{y} = g(t, y(t)), \tag{4.27}$$

where $x, t \in \mathbb{R}$ and y and g are a real valued functions. Intuitively, we have a function that describes the evolution of y with time. In order to find the value of y at each point, we can integrate the function g to measure the change between time points and incrementally obtain a solution. The ODE above with an initial value say $y_0 = y(0)$ is called an *initial value problem.* Mathematically speaking,

$$y_{i+1} = \int_0^{t_{i+1}} \dot{y}\, dt = \int_0^{t_{i+1}} g(t, y(t))\, dt = \underbrace{\int_0^{t_i} g(t, y(t))\, dt}_{y_i} + \int_{t_i}^{t_{i+1}} g(t, y(t))\, dt, \tag{4.28}$$

where y_i denotes the value of $y(t)$ at time t_i, i.e $y_i = y(t_i)$. The problem now reduces to computing the $\int_{t_i}^{t_{i+1}} g(t, y(t))\, dt$ in general interval $[t_i, t_{i+1}]$. In order to compute the integral, we use the approximation for a "small" time step Δt [32],

$$\int_{t_i}^{t_{i+1}} g(t, y(t)) dt \approx g(t, y(t)) \int_{t_i}^{t_{i+1}} dt = g(t, y(t)) \cdot \Delta t. \tag{4.29}$$

If we compute $g(t, y(t))$ at t_i, we obtain the **Euler forward scheme** a.k.a Explicit scheme and if we compute $g(t, y(t))$ at t_{i+1} we obtain the **Euler backward scheme** a.k.a Implicit scheme. The one step solution schemes therefore simplify to

$$\text{Explicit scheme :} \quad y_{i+1} = y_i + \Delta t \cdot g(t_i, y_i). \tag{4.30}$$

$$\text{Implicit scheme :} \quad y_{i+1} = y_i + \Delta t \cdot g(t_{i+1}, y_{i+1}). \tag{4.31}$$

Alternatively, one can also use the Taylor series expansion to derive the above equations [30], which is more common.

4.5.1. Explicit scheme

The explicit scheme is very intuitive to implement because we simply use the slope at the current time step to move to the following time step (equation (4.30)) however, this scheme has conditional stability and is unstable for "large" time steps. We denote the system of

ODEs in the Chaboche Model in equation (4.26) using $S_{ODE} \in \mathbb{R}^3$. The explicit scheme in the vector form is given as,

$$\boldsymbol{Z}_{i+1} := \boldsymbol{Z}_i + \Delta t \cdot S_{ODE}(\boldsymbol{Z}_i), \quad \text{where, } \boldsymbol{Z}_i = \begin{bmatrix} \varepsilon_{xx,i} \\ K_i \\ X_{xx,i} \end{bmatrix}. \tag{4.32}$$

Upon discretising the ODE, we obtain

$$\begin{bmatrix} \varepsilon_{xx,i+1} \\ K_{i+1} \\ X_{xx,i+1} \end{bmatrix} = \begin{bmatrix} \varepsilon_{xx,i} + \Delta t \cdot \left\{ E^{-1} \cdot \dot{\sigma}_{xx,i} + \frac{1}{\sigma_{v,i}} \cdot (\sigma_{xx,i} - X_{xx,i}) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0 \right\} \\ K_i + \Delta t \cdot b_{iso} \cdot (Q_{iso} - K_i) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \\ X_{xx,i} + \Delta t \cdot b_{kin} \left(\frac{2}{3} Q_{kin} \cdot \frac{1}{\sigma_{v,i}} (\sigma_{xx,i} - X_{xx,i}) - X_{xx,i} \right) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \end{bmatrix}. \tag{4.33}$$

We re-write the first equation in the discretised system of ODEs above in terms of stress since it is the dependent quantity in our example and the applied strain is the independent quantity [1]. The first equations re-written in terms of the stress using $\dot{\varepsilon}_{xx,i} = \frac{\varepsilon_{xx,i+1} - \varepsilon_{xx,i}}{\Delta t}$ and $\dot{\sigma}_{xx,i} = \frac{\sigma_{xx,i+1} - \sigma_{xx,i}}{\Delta t}$ are given below,

$$\sigma_{xx,i+1} = \sigma_{xx,i} + \Delta t \cdot E \cdot \left\{ \dot{\varepsilon}_{xx,i} - \frac{1}{\sigma_{v,i}} (\sigma_{xx,i} - X_{xx,i}) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0 \right\}. \tag{4.34}$$

We use the equation above to rewrite the equation (4.33) and the final discretised system for implementation is given in equation (4.35).

$$\begin{bmatrix} \sigma_{xx,i+1} \\ K_{i+1} \\ X_{xx,i+1} \end{bmatrix} = \begin{bmatrix} \sigma_{xx,i} + \Delta t \cdot E \cdot \left\{ \dot{\varepsilon}_{xx,i} - \frac{1}{\sigma_{v,i}} (\sigma_{xx,i} - X_{xx,i}) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0 \right\} \\ K_i + \Delta t \cdot b_{iso} \cdot (Q_{iso} - K_i) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \\ X_{xx,i} + \Delta t \cdot b_{kin} \left(\frac{2}{3} Q_{kin} \cdot \frac{1}{\sigma_{v,i}} (\sigma_{xx,i} - X_{xx,i}) - X_{xx,i} \right) \cdot \left\langle \frac{\sigma_{ex,i}}{D} \right\rangle^n \end{bmatrix}. \tag{4.35}$$

This is fairly straight forward to implement but as mentioned earlier in this section, the explicit scheme is conditionally stable which means that the algorithm must take small steps in order to remain stable which results in longer computational time [30]. This motivates us to consider unconditionally stable schemes such as the Implicit scheme which we will now introduce.

4.5.2. Implicit scheme

The implementation of the implicit scheme is slightly more involved and we shall begin with writing the vector form of equation (4.31). In the implicit scheme, we use the slope at time t_{i+1} to compute $\boldsymbol{Z}_{i+1}$.

$$\boldsymbol{Z}_{i+1} := \boldsymbol{Z}_i + \Delta t \cdot S_{ODE}(\boldsymbol{Z}_{i+1}). \tag{4.36}$$

[1] However, a stress controlled experiment is also valid, in which case the system of equations would remain unchanged.

Since the term $\boldsymbol{Z}_{i+1}$ appears on both sides of the equation,the equation is clearly implicit. This system of equations $S_{EQ}(\boldsymbol{Z}_{i+1}) \in \mathbb{R}^3$ can be solved using the *Newton-Rahpson scheme*. This is the computational cost of the unconditional stability the algorithm provides. This additional cost is however not very large and is usually offset by the stability it provides for large time steps especially when the ODE system is *stiff* [30].

$$S_{EQ}(\boldsymbol{Z}_{i+1}) := \boldsymbol{Z}_{i+1} - \boldsymbol{Z}_i - \Delta t \cdot S_{ODE}(\boldsymbol{Z}_{i+1}) = 0. \tag{4.37}$$

Upon discretising the system of ODEs as above, we obtain S_{EQ} for the $1D$ Chaboche model as follows:

$$S_{EQ} = \begin{bmatrix} F \\ g \\ H \end{bmatrix} =$$

$$\begin{bmatrix} \varepsilon_{xx,i+1} - \varepsilon_{xx,i} - \Delta t \cdot \left\{ E^{-1} \cdot \left(\frac{\sigma_{xx,i+1} - \sigma_{xx,i}}{\Delta t} \right) + \frac{1}{\sigma_{v,i+1}} (\sigma_{xx,i+1} - X_{xx,i+1}) \cdot \left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0 \right\} \\ K_{i+1} - K_i - \Delta t \cdot b_{iso} \cdot (Q_{iso} - K_{i+1}) \cdot \left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n \\ X_{xx,i+1} - X_{xx,i} - \Delta t \cdot b_{kin} \left(\frac{2}{3} Q_{kin} \cdot \frac{1}{\sigma_{v,i+1}} (\sigma_{xx,i+1} - X_{xx,i+1}) - X_{xx,i+1} \right) \cdot \left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n \end{bmatrix}.$$

Newton-Raphson scheme

The Newton-Raphson(NR) scheme is a root-finding algorithm to iteratively find the solution to a (system) of non-linear equation(s). Consider a non-linear equation $g(x)$ with a scalar variable $x \in \mathbb{R}$ which has continuous derivative g'. In order to find the root, we first consider the Taylor series expansion around a given point [18].

$$g(x + \delta x) = g(x) + \sum_{s=1}^{r} \frac{1}{s!} \cdot g^{(s)}(x) \cdot \delta(x)^s + R. \tag{4.38}$$

We truncate the expansion (4.38) and consider only the first term in the summation.

$$g(x + \delta x) = g(x) + g'(x) \cdot \delta x. \tag{4.39}$$

We now assume the function value to be zero at the following step and set $g(x + \delta x) = 0$. Additionally, we also substitute $\delta x = x^{j+1} - x^j$ and rewrite (4.39) as follows,

$$x^{j+1} = x^j - \frac{g(x^j)}{g'(x^j)} \quad \text{with} \quad g'(x^j) \neq 0. \tag{4.40}$$

We can now iteratively search for the root starting from an initial guess $x^{(0)}$ until some convergence criteria is met. The NR scheme converges quickly when the initial guess is close to the root. In our case, we employ the NR scheme in each iteration of the implicit scheme and therefore providing a good initial guess becomes necessary to speed up computations. It is common to perform one explicit step to provide an initial guess for the

NR scheme [p.324 30]. This generally provides a good starting point and the computational cost is minimal.

We now write down the NR scheme for the system of non-linear equations (4.37). The index i in the subscript refers to the iteration in the implicit scheme and the index j in the superscript refers to the NR iterations. As opposed to the derivative g' in the scalar case (4.40), the term becomes a *Jacobian* in the vectorial case which must be inverted [2].

$$\boldsymbol{Z}^{j+1} := \boldsymbol{Z}^{j} - \boldsymbol{J}_{S_{EQ}}^{-1}(\boldsymbol{Z}^{j}) \cdot \boldsymbol{S}_{EQ}(\boldsymbol{Z}^{j}). \tag{4.41}$$

The jacobian $\boldsymbol{J}_{S_{EQ}}[3 \times 3]$ for the 1D model is given below,

$$\boldsymbol{J}_{S_{EQ}}(\boldsymbol{Z}_i) := \begin{bmatrix} \frac{\partial F}{\partial \sigma_{xx,i+1}} & \frac{\partial F}{\partial K_{i+1}} & \frac{\partial F}{\partial X_{xx,i+1}} \\ \frac{\partial g}{\partial \sigma_{xx,i+1}} & \frac{\partial g}{\partial K_{i+1}} & \frac{\partial g}{\partial X_{xx,i+1}} \\ \frac{\partial H}{\partial \sigma_{xx,i+1}} & \frac{\partial H}{\partial K_{i+1}} & \frac{\partial H}{\partial X_{xx,i+1}} \end{bmatrix}_{3\times 3} . \tag{4.42}$$

The derivation of the individuals terms of the Jacobian can be found in Appendix A.

4.6. Example: 1D Monotonic Loading

In order to demonstrate the viscoplastic behaviour modeled by the Chaboche model, we shall compute the stress strain curve of a 1D specimen subjected to monotonic loading using sample material parameter values from [17]. The values correspond to structural steel $S355$ typically used in construction.

Parameter	E	ν	$\sigma_{y\|0}$	n	D	Q_{iso}	b_{iso}
Value	200000	0.3	250	2	100	50	100
Unit	N/mm^2	-	N/mm^2	-	N/mm^2	N/mm^2	-

Table 4.1.: Parameters for 1D monotonic loading example

The monotonic loading applied is plotted in Figure 4.4. Here, a strain rate of $5 \times 10^{-2}[1/s]$ is applied for $1s$ starting from 0 and is held there for $1s$ once the maximum strain is reached. The material response is computed using the implicit scheme using the parameters in table 4.1 with a time step $\Delta t = 1 \times 10^{-3}$ which requires $0.5s$ to compute. The computations are validated by comparing it against the explicit scheme with a time step of $\Delta t = 1 \times 10^{-4}$.

[2] Technically, for large systems, such as the 2D/3D cases, it is expensive to directly invert the matrix and instead a system of linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is solved iteratively, where $\boldsymbol{A} = \boldsymbol{J}_{S_{EQ}}(\boldsymbol{Z}^{j})$ and $\boldsymbol{b} = \boldsymbol{S}_{EQ}(\boldsymbol{Z}^{j})$.

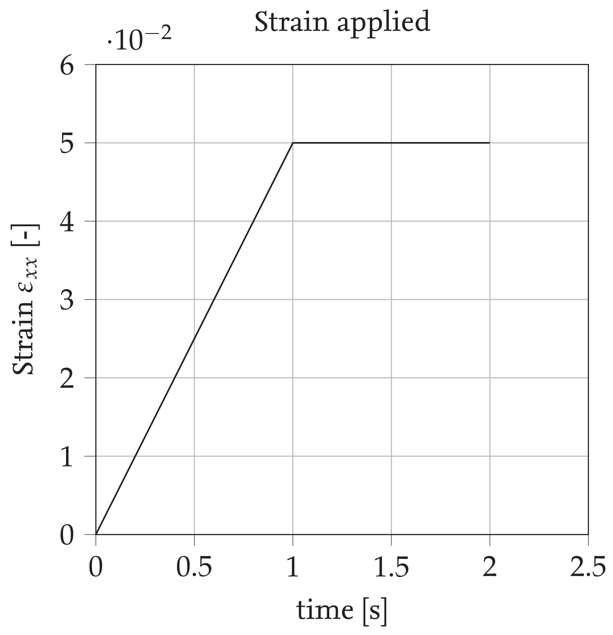


Figure 4.4.: Strain applied in the 1D monotonic loading example.

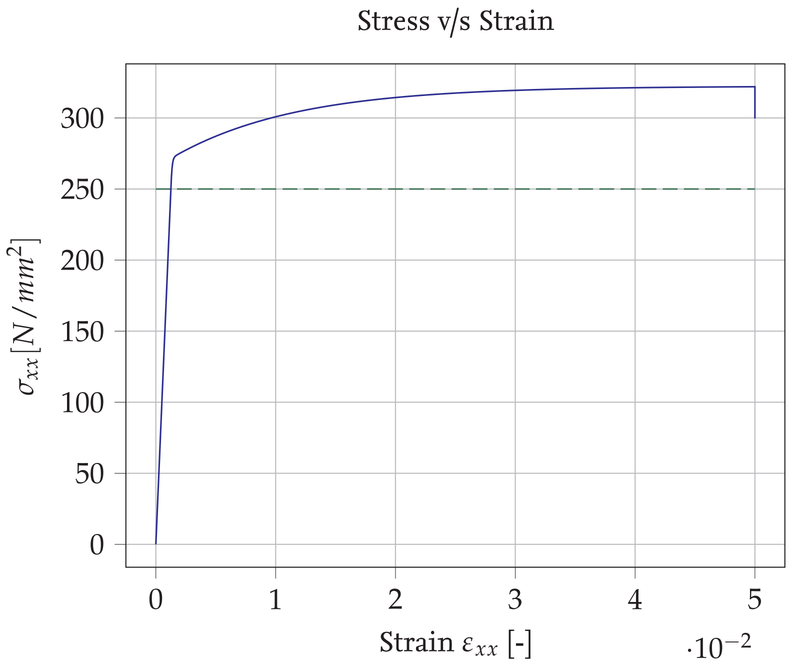


Figure 4.5.: Stress Strain curve obtained from the Chaboche model in the 1D monotonic loading example. The horizontal green line denotes the initial yield limit $\sigma_{y|0} = 250$ MPa.

The resulting stress-strain response of the model is plotted in Figure 4.5. In the Figure, the elastic region ($\sigma_{xx} \leq \sigma_{y|0}$) and elasto-plastic region ($\sigma_{xx} > \sigma_{y|0}$) are highlighted using the dashed green line. Although plastic strains appear once the stresses exceed the yield limit, the beginning of the non-linearity (the first kink in the curve) does not begin until slightly later. This can be explained as follows, the plastic multiplier term $|\dot{\varepsilon}^{pl}| = \dot{\lambda} = \left\langle \frac{\sigma_{ex}}{D} \right\rangle^n \cdot \dot{\varepsilon}_0$ (governed by parameters D and n) evolves slowly and the elastic strain rate primarily influences the evolution of the stress curve $\dot{\sigma} = E(\dot{\varepsilon}^{el}) = E(\dot{\varepsilon} - \dot{\varepsilon}^{pl})$ (equation (4.2)). Once the plastic strains overtake the elastic strains, the non-linearity begin to appear. In order to illustrate this, the plastic multiplier is plotted along with the stress in Figure 4.6. We note that the first jump in the plastic multiplier (in red) coincides with the beginning of the non-linearity.

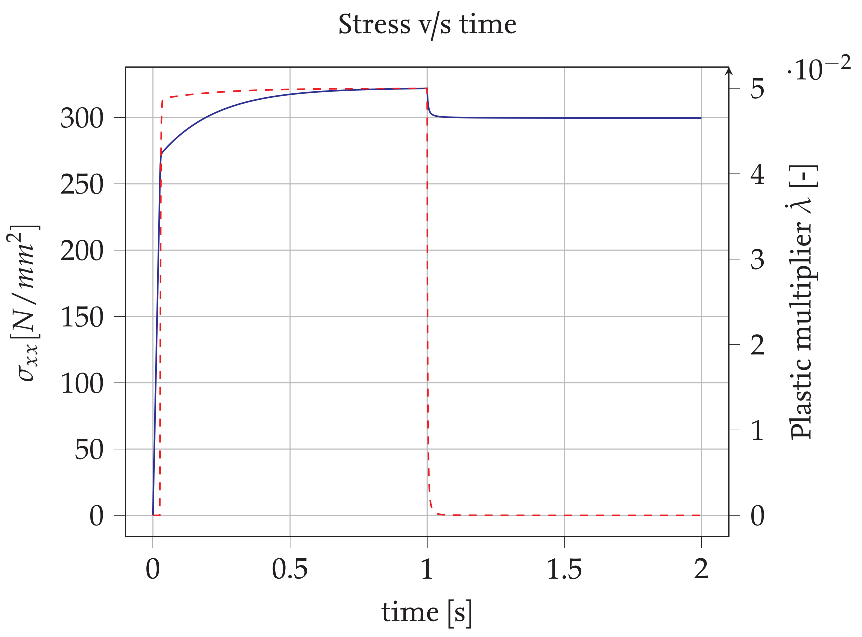


Figure 4.6.: Evolution of stress (in blue) and plastic multiplier (in red) in the 1D monotonic loading example.

At the end of 1s, the strain is held constant at 5×10^{-2} which causes relaxation in the specimen, i.e. the stress σ_{xx} in the specimen decreases despite a constant strain. During relaxation the excess stress σ_{ex} asymptotically returns to 0 (see plastic multiplier curve in red in Figure 4.6). This implies that the stress in the specimen σ_{xx} asymptotically approaches the yield limit σ_y [17]. Therefore we observe in Figure 4.6 that the stress approaches the yield limit $\sigma_y = 300\text{N/mm}^2$.

In order to understand how the yield limit changed from $\sigma_{y|0} = 250\text{N/mm}^2$ to $\sigma_y = 300\text{N/mm}^2$, consider the isotropic hardening equation

$$\dot{K} = b_{iso} \cdot (Q_{iso} - K) \cdot \left\langle \frac{\sigma_{ex}}{D} \right\rangle^n = 0.$$

The isotropic hardening continues to evolve until either the saturation value is reached, i.e. $K = Q_{iso}$ or $\sigma_{ex} = 0$. This implies that the isotropic hardening can continue to evolve even at a constant strain if one of the two previously mentioned conditions is met [18]. This is illustrated in Figure 4.7, where the isotropic hardening asymptotically reaches 50N/mm^2. Mathematically,

$$\lim_{t\to\infty} \sigma_y = \lim_{t\to\infty} \left(\sigma_{y|0} + K\right) = \sigma_{y|0} + \lim_{t\to\infty} K = 250\,\text{N/mm}^2 + 50\,\text{N/mm}^2. \tag{4.43}$$

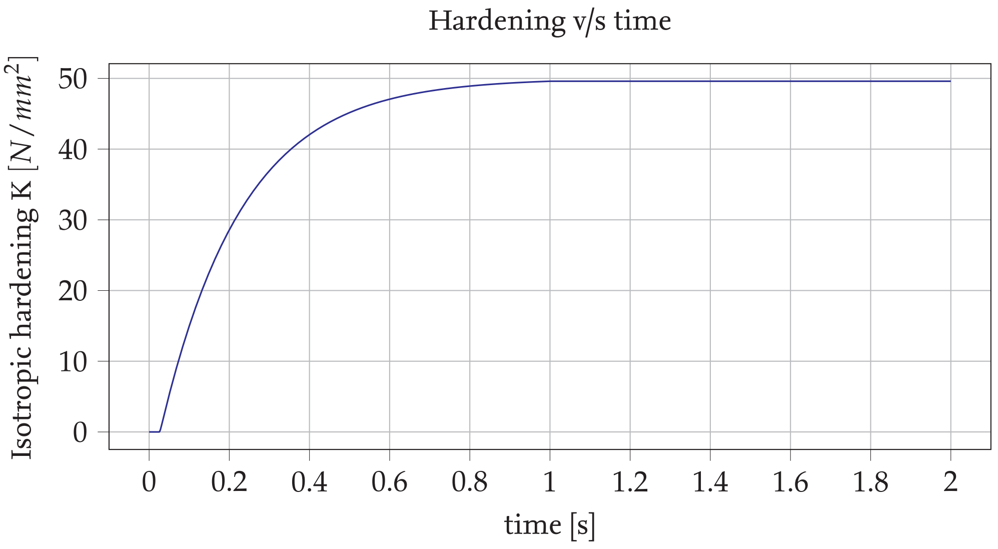


Figure 4.7.: Evolution of isotropic hardening in the 1D monotonic loading example.

The isotropic hardening parameter in general however may not always reach the asymptotic value Q_{iso} as in this example and might only reach a lower value if the parameter b_{iso} is "small". In this example, if we only change the value of $b_{iso} = 100$, to say $b_{iso} = 20$, K will only reach a maximum value of 31.072N/mm^2 and the final yield limit will be $\sigma_y = 250 + 31.072 = 281.072$N/mm^2.

4.7. Example: 1D Cyclic Loading

Parameter	E	ν	$\sigma_{y\|0}$	n	D	Q_{iso}	b_{iso}	Q_{kin}	b_{kin}
Value	200000	0.3	250	2	100	50	100	75	30
Unit	N/mm^2	-	N/mm^2	-	N/mm^2	N/mm^2	-	N/mm^2	-

Table 4.2.: Parameters for 1D cyclic loading example

In the previous example, we subjected the specimen to monotonic loading and observed the isotropic hardening. To observe the kinematic hardening however we must subject the specimen to cyclic loading so that both effects can be studied.

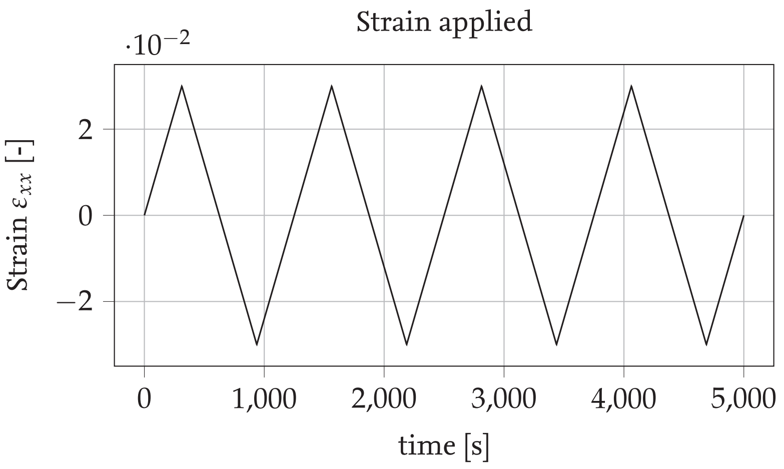


Figure 4.8.: Strain applied in the 1D cyclic loading example.

The monotonic loading applied is plotted in Figure 4.8. The maximum strain in the loading is 3×10^{-2} in 312.5s which results in a strain rate $\dot{\varepsilon}_{xx} = \frac{3\times10^{-2}}{312.5} = 9.6 \times 10^{-5}[1/s]$. Since the strain rate is very low, the viscoplastic effects have a negligible effect and the process can be assumed quasi-static [17]. The problem is solved using the implicit scheme as before but with a time step $\Delta t = 2s$ with the parameters from table 4.2. The computation required approximately 3.2s.

The resulting stress-strain curve is plotted in 4.9 and matches example reproduced from [17]. Additionally we also plot the evolution of stress in Figure 4.10 with the applied strain (in red) superimposed to highlight effect of load reversal. We note that the stress quickly changes the direction but reaches the maximum only incrementally.

The evolution of the isotropic and kinematic hardening are shown in Figure 4.11 and here once again, the applied strain (in red) superimposed to highlight effect of load reversal on the back stress X_{xx}. The isotropic hardening asymptotically reaches its saturation point $Q_{iso} = 50\text{N/mm}^2$ but the back stress X_{xx} does not reach the asymptotic value of $\frac{2}{3}Q_{kin} = 50\text{N/mm}^2$ and only reaches a maximum value of 34.46N/mm^2 [as in 17].

The results are in good agreement with the values stated in [17] which validates our implementation. We can now therefore employ this model for sensitivity studies.

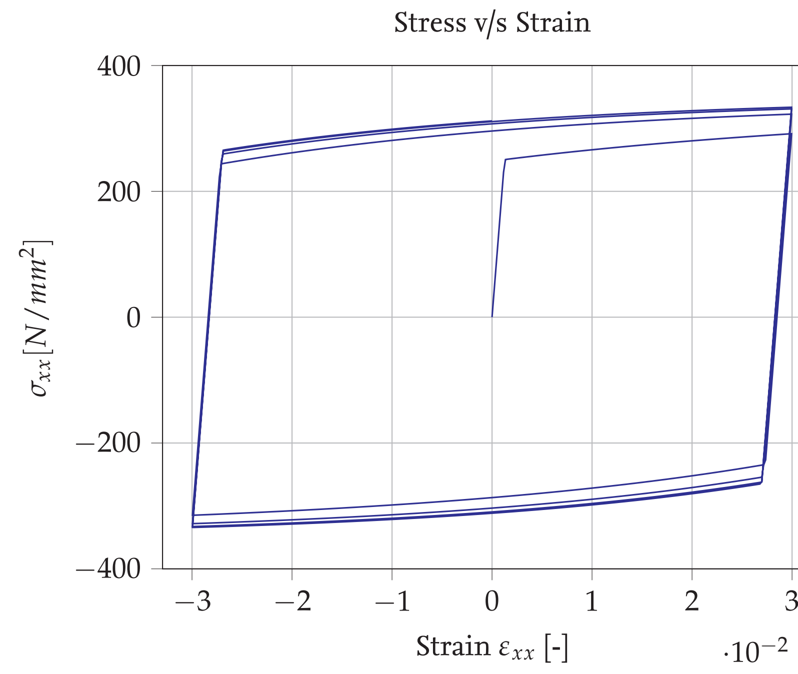


Figure 4.9.: Stress Strain curve obtained from the Chaboche model in the 1D cyclic loading example.

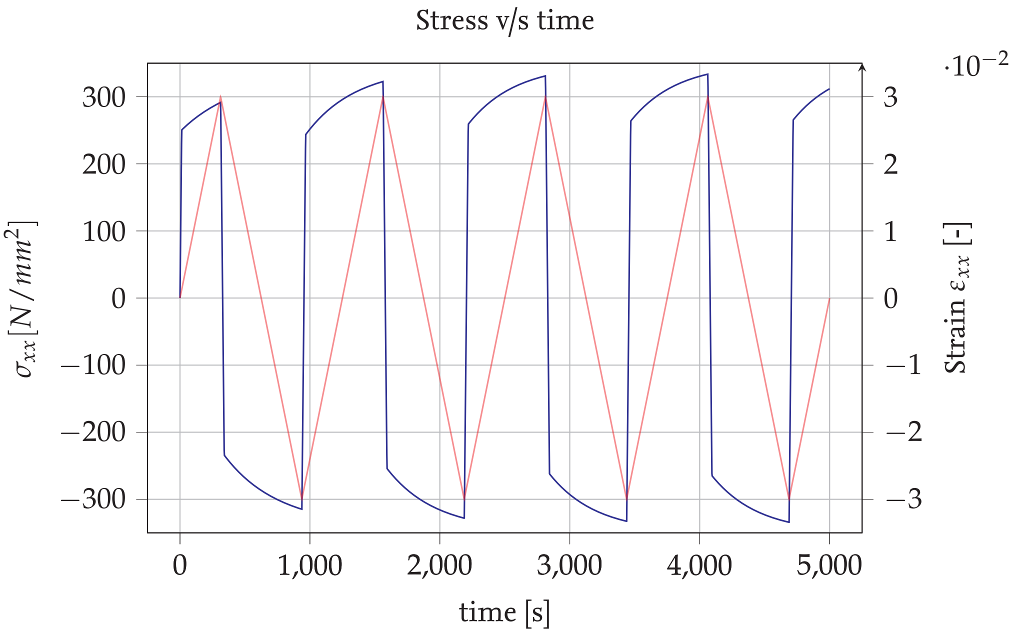


Figure 4.10.: Evolution of stress in the 1D cyclic loading example. Applied strain (in red) superimposed to highlight effect of load reversal.

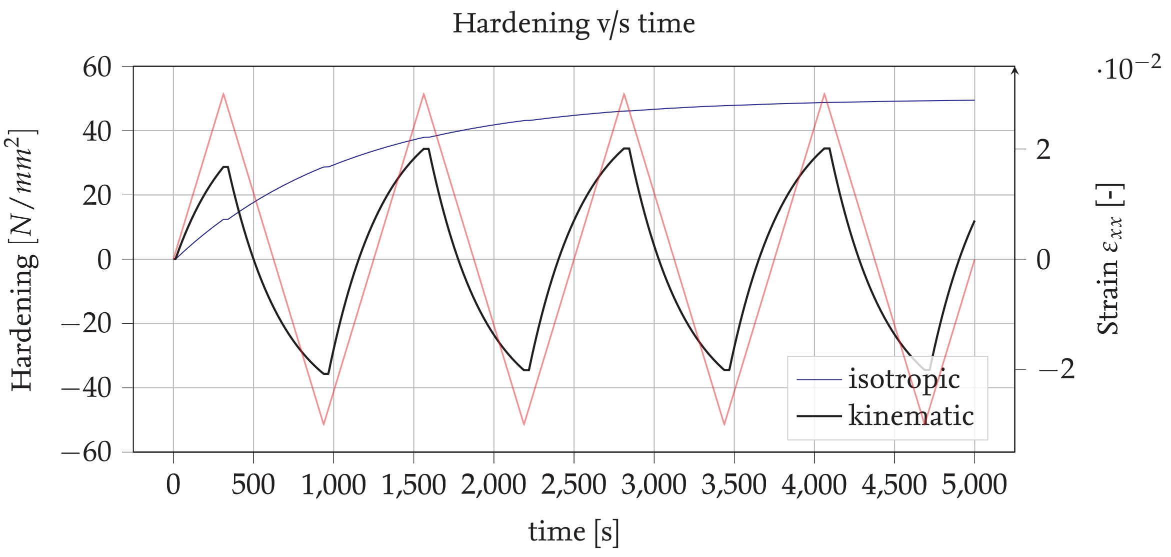


Figure 4.11.: Evolution of isotropic and kinematic hardening in the 1D cyclic loading example. Applied strain (in red) superimposed to highlight effect of load reversal.

5. Results

In this chapter, we present the results of the sensitivity analysis of the Chaboche model computed using the techniques introduced in chapter 3. Computation of the indices using Monte Carlo techniques turns out to be intractable due to the large number of model evaluations necessary. In order to compute the indices efficiently, we first construct time-frozen PCE and KLE+PCE surrogate models. Finally, using the insights from the sensitivity analysis, we try to carry out an elementary bayesian calibration of the model and discuss the obstacles encountered in the calibration and the usage of surrogate models therein.

5.1. Physical model setup

In this section, we describe the setup used for computing the viscoplastic response of the 1D Chaboche model to conduct UQ studies. In this work, we restrict our focus to the monotonic loading case with a strain rate $\dot{\varepsilon} = 5 \times 10^{-2}[1/s]$ as in the example in section 4.6. The parameters Q_{kin} and b_{kin} corresponding to the kinematic hardening can be neglected. Therefore our model is now,

$$\mathcal{M}(t, \xi) = \sigma_{xx}(t, E, \sigma_{y_0}, Q_{iso}, b_{iso}, n, D). \tag{5.1}$$

The parameters in equation (5.1) are modeled using uniform random variables and their respective bounds are given in Table 5.1.

#	ξ	**Distribution**	**a**	**b**	**Mean**	**Variance**	**Unit**
1	E	Uniform	$200,000$	$210,000$	$205,000$	$8,333,333.33$	N/mm^2
2	σ_{y_0}	Uniform	200	400	300	$3,333.33$	N/mm^2
3	Q_{iso}	Uniform	0	500	250	$20,833.33$	N/mm^2
4	b_{iso}	Uniform	0	1000	500	$83,333.33$	-
5	n	Uniform	1	6	3.5	2.08	-
6	D	Uniform	1	100	50.5	816.75	N/mm^2
7	Q_{kin}	Uniform	0	500	250	$20,833.33$	N/mm^2
8	b_{kin}	Uniform	0	1000	500	$83,333.33$	-

Table 5.1.: Chaboche model parameters and their respective probability distributions [17].

To visualise the influence of the parameters, we plot 50 realisations of the model in Figure 5.1 using the random variables from Table 5.1.

In Figure 5.2, we plot the pointwise variance in time of the model computed using $N = 5,000$ MC samples. From the Figure, it is clear that the variance is significantly larger

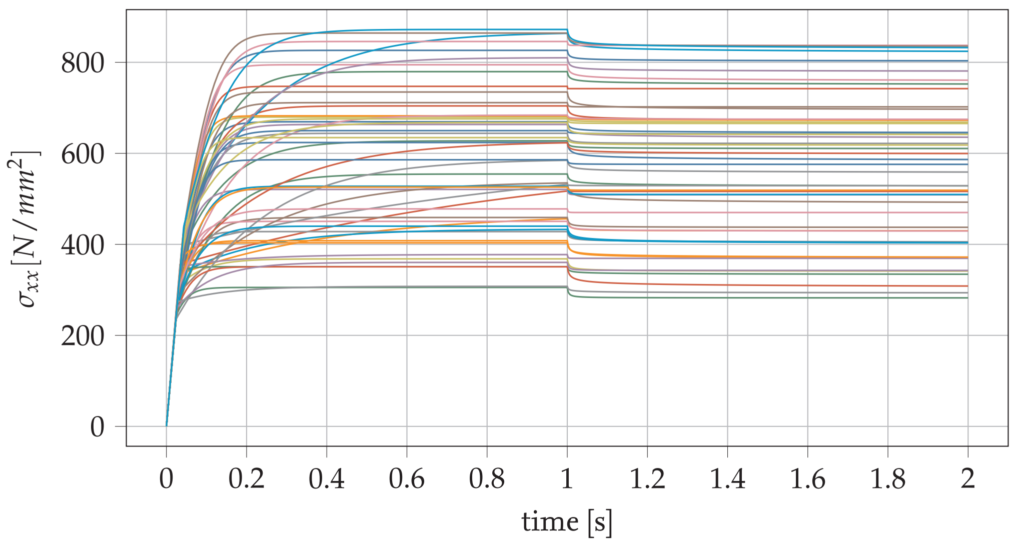


Figure 5.1.: 50 realisations of the Chaboche model

in the non-linear regime of the output compared to the initial linear regime, which is consistent with the plot of realisations in the above Figure.

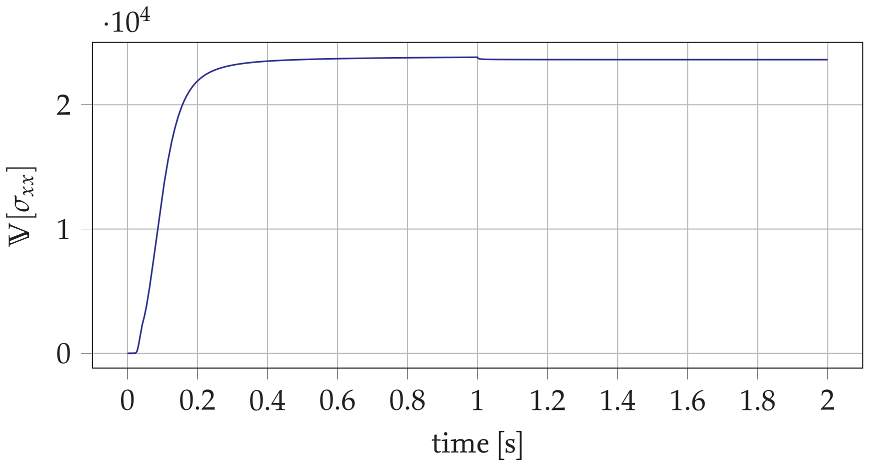


Figure 5.2.: Pointwise variance over time

In order to accelerate the model evaluations necessary for training the surrogates, the Chaboche model solver is parallelised. This significantly improves the training times in the monotonic loading case where a single model evaluation requires $\sim$ 0.2s (and will be even more beneficial for the cyclic loading case where a single evaluation requires $\sim$ 3.2s).

5.2. Surrogate models

In this section, we build the two surrogate models, time-frozen PCE and KLE+PCE, introduced in 2.1.6 and 2.2 respectively.

5.2.1. Time-frozen PCE surrogates

In this section, we will construct time-frozen PCE surrogates as illustrated in section 2.1.6. Since the variance at $t = 0$ is 0, we construct 100 surrogates at equidistant points in the time interval $[0.01, 2]$, i.e at $t_i \in \{0.01, 0.02, \dots 2\}$s. Each surrogate $\tilde{\sigma}_{xx}^{PCE}(t_i, \boldsymbol{\xi})$ is constructed using the Legendre family of orthogonal polynomials since the germ is a uniform random variable. The maximum total degree of the polynomials is $p = 6$, which leads to a polynomial basis with $N_{PC} = \frac{(N_p+p)!}{N_p!p!} = \frac{(6+6)!}{6!6!} = 924$ terms. Empirically, at least $N_{PC}(N_p - 1) = 924 * 5 = 4,620$ model evaluations are necessary to train the surrogate [28]. Therefore, we use $N = 5,000$ MC samples to compute the model responses for training. During training we also compute the LOO error for each surrogate and plot it in Figure 5.3.

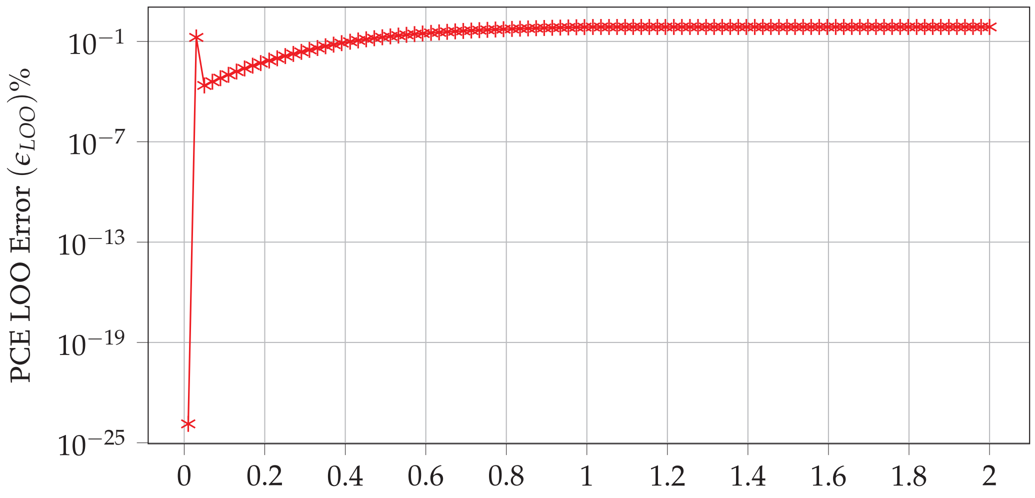


Figure 5.3.: LOO error of 100 time-frozen PCE surrogates.

The sudden jump in the beginning in the error curve in Figure 5.3 can be explained as follows: The variance in the initial elastic region is solely governed by the Young's Modulus and therefore the PCE error at $t = 0.01$s is extremely low ($\mathcal{O}(10^{-25})$), since the dependence of the model w.r.t. Young's modulus is easy to approximate using polynomials. The first instance of plasticity begins at $t = \frac{\sigma_{y0}}{E \cdot \dot{\varepsilon}} = \frac{200}{2.1\times10^5 \cdot 5\times10^{-2}} \approx 0.019$s, for the set of parameters with the smallest $\sigma_{y|0}$ and the largest E, which makes the model non-linear and therefore the PCE error at $t = 0.02$s is much higher. Since we use a constant p for all time-frozen PCE surrogates and the error increases over time, it can concluded that the underlying distribution becomes increasingly more difficult to approximate. The maximum LOO error is 0.781% at $t = 2$s.

Although the PCE error in the time-frozen surrogates increases, in this case it is reasonably small and can be employed for various tasks in UQ (this may however change in the cyclic loading case, due the complexity of the process and the time range for which it is considered).

5.2.2. KLE+PCE surrogates

As outlined in section 2.2, we construct a global surrogate $\tilde{\sigma}_{xx}^{KL+PC}(t, \boldsymbol{\xi})$ for the monotonic loading response of the Chaboche model. Unlike the time-frozen surrogate, the KLE+PCE surrogate is trained using $N = 10^4$ MC samples, where each evaluation generates solution at 2001 points in the interval $[0, 2]$s. We begin by computing the eigenvalues of the covariance operator and plot them in Figure 5.4. Since the process is low-rank, we construct an approximation using $N_{kl} = 10$ basis terms which explains 99.9996% of the variance. The resulting projections of the process on the eigenbasis are approximated using gPCE surrogates of total polynomial degree $p = 7$, which empirically requires $8,580$ function evaluations. The resulting error ϵ_{LOO} in the PCE approximation of the modes is plotted in Figure 5.5.

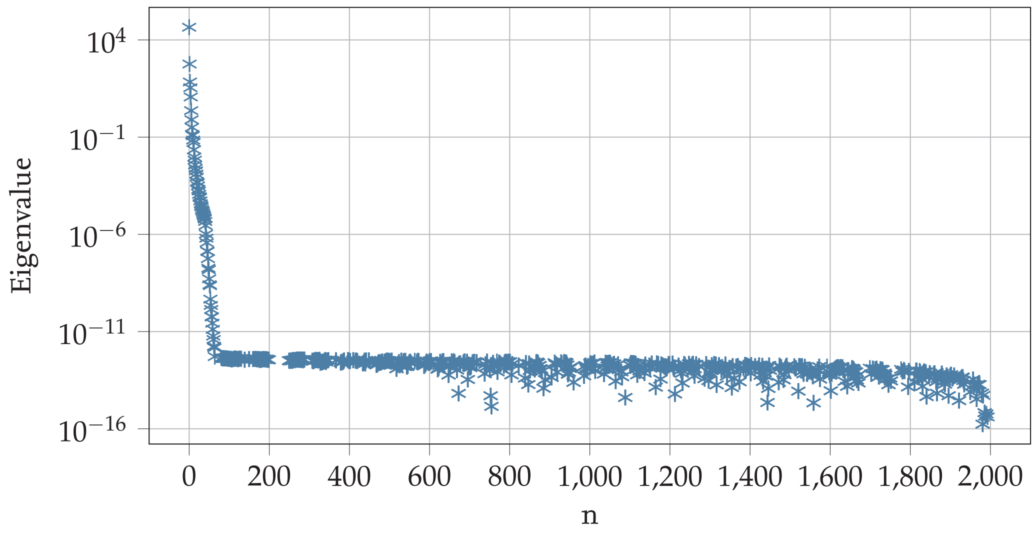


Figure 5.4.: Eigenvalues of the covariance operator of the monotonic response of the Chaboche model.

Similar to the mechanical oscillator example, we plot the error in the pointwise variance (in red) approximated by the surrogate in Figure 5.6. It is clear that the initial area with low variance is poorly approximated with an error close to 50%.

We compute the surrogate error using relation (2.51),

$$\epsilon_{KL+PC} = \left(\sqrt{\epsilon_{N_{kl}}} + \sqrt{\epsilon_{LOO,KL}}\right)^2 * 100 \quad = \left(\sqrt{3.85e-06} + \sqrt{7.26e-06}\right)^2 * 100 = 0.0022\%.$$

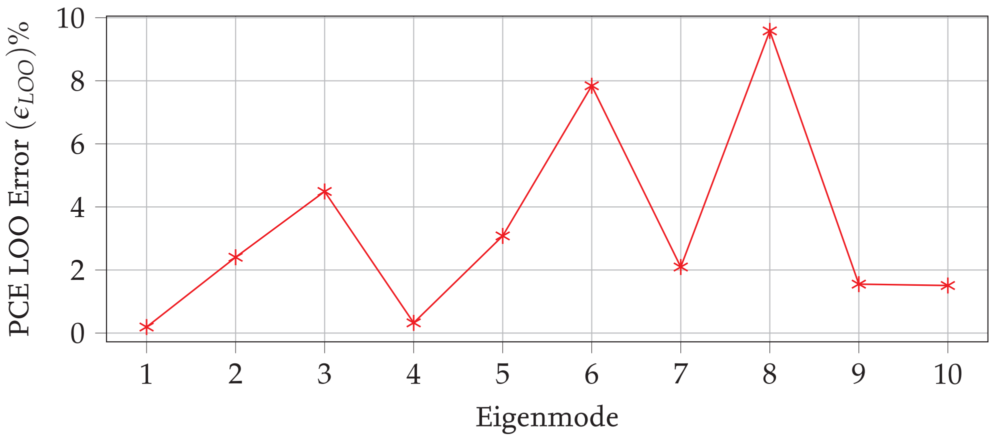


Figure 5.5.: LOO error of the PCE approximations of the projections of the model response on the eigenbasis.

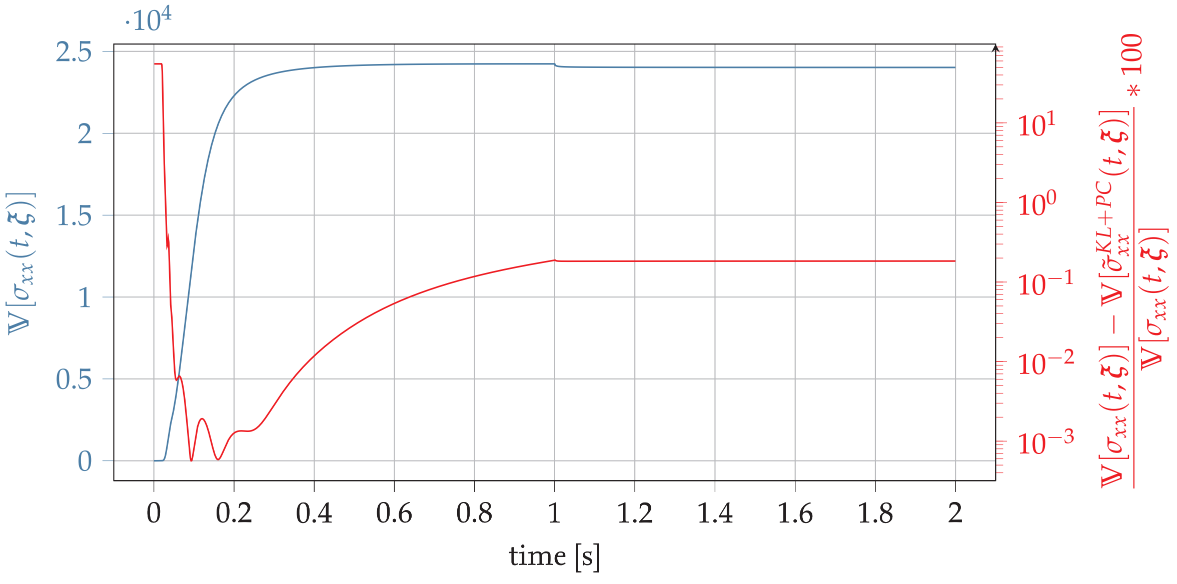


Figure 5.6.: Pointwise error in the variance approximated by the KLE+PCE surrogate (in red) and the estimated pointwise variance (in blue) using model evaluations with MC samples

The error is much lower than 5% as suggested by Wagner et al. in [34].

5.3. Sensitivity Analysis

We now carry out sensitivity analysis of the monotonic loading response of the Chaboche model to understand the contributions of the 6 parameters towards the uncertainty. As introduced in section 3, we shall compute the pointwise-in-time Sobol indices and the generalised Sobol indices and interpret the results.

5.3.1. Pointwise-in-time Sobol indices

The pointwise-in-time Sobol indices are computed using 3 methods,

1. **Pick and Freeze Estimator**: To compute the sensitivities, we use $N = 10^4$ MC samples which require $N(N_p + 2) = 8 \times 10^4$ model evaluations. Although the large computation time is made tractable by the parallel implementation, this method is also memory intensive since all the computed evaluations must be stored to evaluate the sensitivities. In this experiment, the sensitivity indices **did not converge** (they exceed the range $[0, 1]$) and clearly require a larger number of MC samples (atleast of order 10^5) to converge or a more efficient estimator must be used. Refer Figure B.1.

2. **PCE surrogate**: In order to circumvent the slow convergence in estimating the indices using the Pick and Freeze estimator, we compute pointwise-in-time PCE surrogates as given in section 5.2.1. Once the surrogate is computed, the Sobol indices can be directly estimated. We plot the First-order and Total-order Sobol indices in Figure 5.7 and Figure 5.8 respectively.

3. **KLE+PCE surrogate**: Using the KLE+PCE surrogate from section 5.2.2, we replace the model evaluations in the Pick and Freeze estimator. Since the surrogate is computationally inexpensive to evaluate, we use $N = 10^5$ MC samples. The resulting indices are plotted in Figure B.2 and Figure B.3. We note that the Sobol indices in the region of low variance **do not converge** since the surrogate approximation in that region is poor. The estimates of the indices post that region seem to in agreement with the PCE surrogate.

From Figure 5.7 and 5.8, we can infer that the Young's modulus E is solely responsible for the output uncertainty in the elastic region and has a negligible sensitivity in the rest of the process. There are two other peaks in the plot of parameter $\sigma_{y|0} = 0.916$ at $t = 0.03$s and $b_{iso} = 0.301$ at $t = 0.09$s. This implies that these parameters make a large contributions towards the model output uncertainty at those time points. Next, we study the pointwise interaction effects of the parameters $\mathcal{S}_{T_i} - \mathcal{S}_i$ and plot them in Figure 5.9.

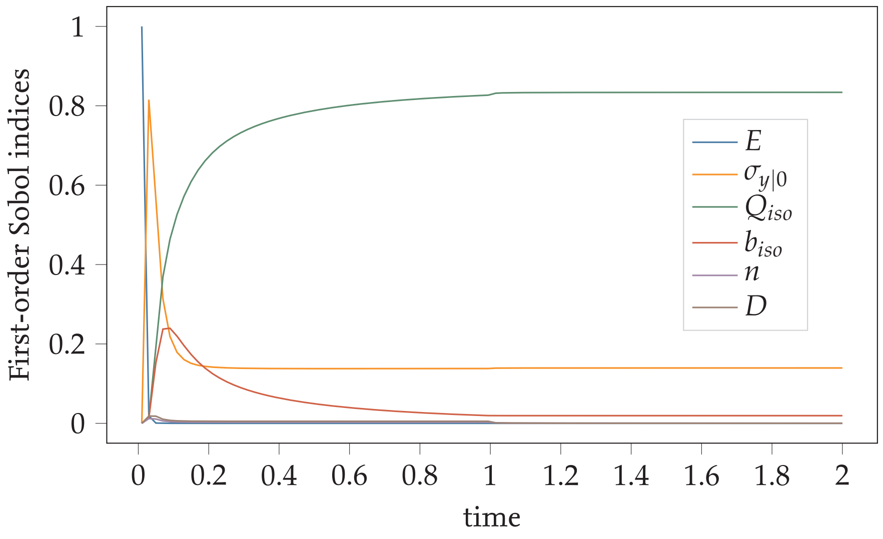


Figure 5.7.: Pointwise-in-time First-order Sobol indices of the Chaboche model computed using a gPCE surrogate.

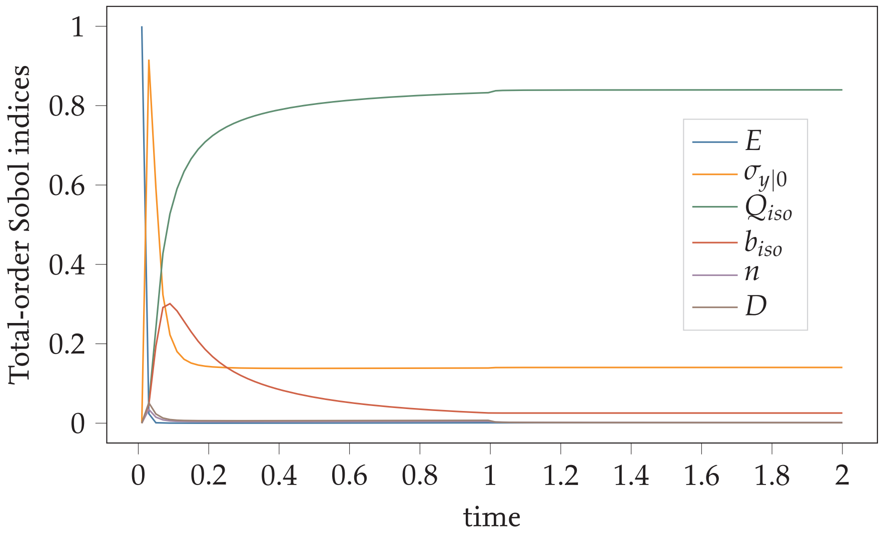


Figure 5.8.: Pointwise-in-time Total-order Sobol indices of the Chaboche model computed using a gPCE surrogate.

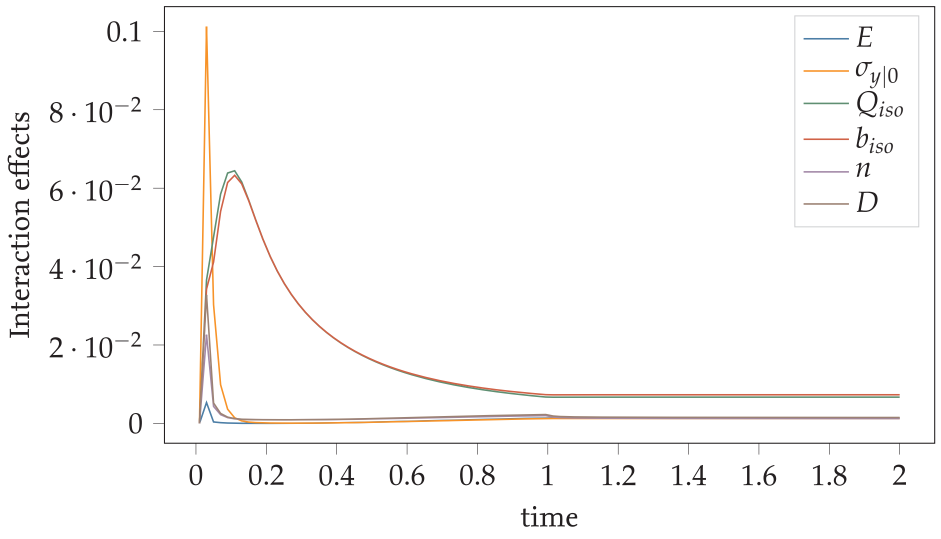


Figure 5.9.: Pointwise-in-time interaction effects of the parameters of the Chaboche model computed using a gPCE surrogate.

Overall, the interactions effects seem to be weak as the order of the values is small. However, the interaction effects of the parameters peak simultaneously at two time points and are tabulated in Table 5.2. The parameters E, σ_{y_0}, n, D peak simultaneously at $t = 0.03$s. The parameter E influences the elastic strains, i.e. the elastic region, σ_{y_0} determines the limit of the elastic region and the parameters n, D appear in the plastic multiplier term in (4.15), which governs the beginning of the non-linearity post the linear regime. The parameters Q_{iso}, b_{iso} govern the isotropic hardening and occur in the same expression in the relation (4.21).

X	E	σ_{y_0}	Q_{iso}	b_{iso}	n	D
$(\mathcal{S}_{T_i} - \mathcal{S}_i)_{max}$	0.005	0.101	0.064	0.064	0.023	0.033
t [s]	0.03	0.03	0.11	0.11	0.03	0.03

Table 5.2.: Interaction effects of the Chaboche model parameters

5.3.2. Generalised Sobol indices

In order to obtain a holistic and consistent estimate of the sensitivities of a random process, we compute the generalised Sobol indices as illustrated for the mechanical oscillator in section 3.2.3.

We compute the evolution of the GSI using the PCE surrogate from the previous section and plot the resulting First-order and Total-order generalised Sobol indices curves in Figure 5.10 and 5.11 respectively. We note that the evolution plot is not very different from the pointwise-in-time Figures because the pointwise variance in time (refer Figure

5.2) and Sobol indices do not vary significantly over time. In case of the mechanical oscialltor example, both the variance in Figure 3.4 and the Sobol indices in Figure 3.2 vary quickly over time, which resulted in a large contrast between the pointwise-in-time and the generalised Sobol indices.

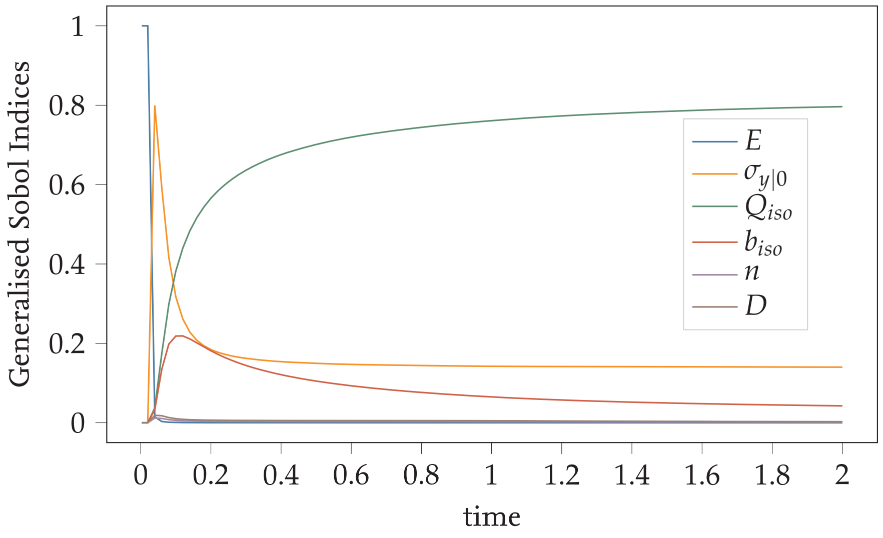


Figure 5.10.: Evolution of generalised First-order Sobol indices of the Chaboche model computed using a gPCE surrogate.

Additionally, we compute the GSI using the KLE+PCE surrogate at $t = 2$s and plot it in Figures 5.12 and 5.13 alongside the indices computed using the PCE surrogate. The results from both approaches appear to be consistent. It is interesting to note that the KLE+PCE surrogate provides a good estimate of the GSI but performs poorly in the pointwise Sobol indices. This can be explained as follows: since the GSI use the total variance of the process to estimate the sensitivities, the KLE+PCE surrogate performs well since it can approximate the total variance well and only performs poorly in areas of low variance.

5.4. Bayesian Calibration

Previously we dealt with the forward propagation of uncertainties in a model, where given a model

$$\mathcal{M}(t, \xi) : \mathcal{T} \times \mathbb{R}^{N_p} \to \mathbb{R},$$

with input parameters ξ with support $\mathcal{D}_\xi$, we were interested in the output uncertainty. In this chapter we briefly discuss another important problem area in UQ called model calibration. In this setting we are interested in inferring the model parameters ξ, given finitely many observations of the model output $\mathcal{Y} := \{y_1, \ldots, y_N\}, y_i \in \mathbb{R}^m$, where m denotes the discretisation in time and N denotes the number of observations. However, the

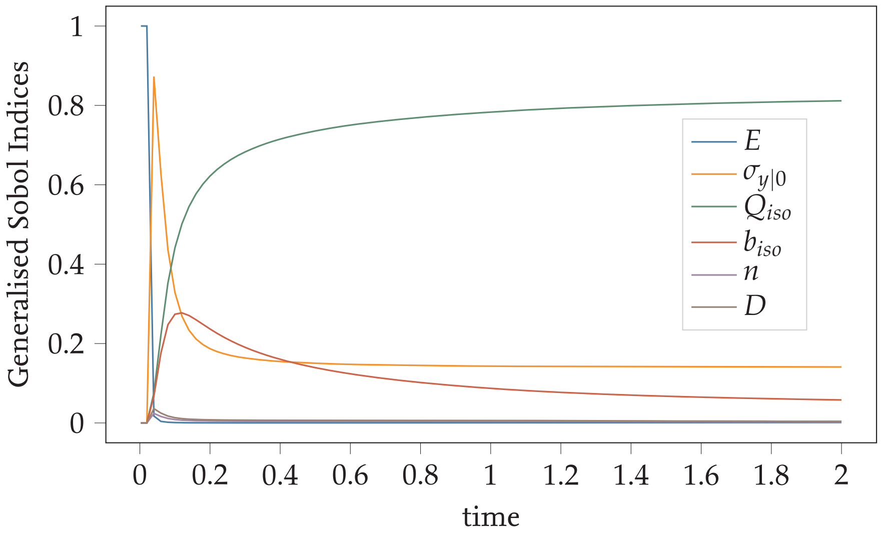


Figure 5.11.: Evolution of generalised Total-order Sobol indices of the Chaboche model computed using a gPCE surrogate.

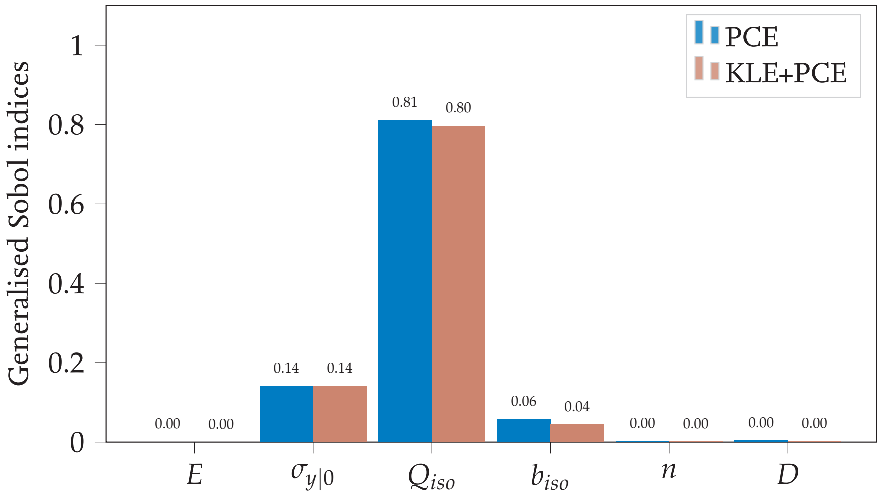


Figure 5.12.: First-order generalised Sobol indices at $t = 2$s computed using two approaches.

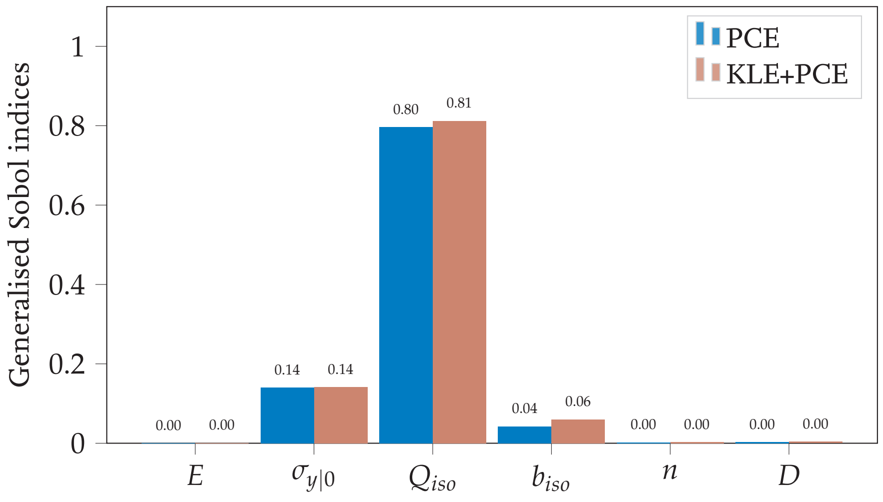


Figure 5.13.: Total-order generalised Sobol indices at $t = 2$s computed using two approaches.

observations made in reality deviate from the model predictions for two reasons: models are approximations of reality (model inaccuracy) and measurements made in the real world are error prone (measurement errors). We therefore write the model observations and predictions as follows [35],

$$y = \mathcal{M}(t, \xi) + \delta, \tag{5.2}$$

where $\delta \in \mathbb{R}^m$ combines the above errors and is called the *discrepancy* term. For simplicity, we assume an additive Gaussian discrepancy:

$$\delta \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \tag{5.3}$$

with mean $\mathbf{0}$ and covariance $\mathbf{\Sigma}$.

There are primarily two approaches to solving the inverse problem: the *frequentist* perspective and the *Bayesian* approach. The frequentist perspective assumes that the model has "determinstic" input parameters which can be determined using only $\mathcal{Y}$ and the error in the parameter determination stems from δ. The Bayesian perspective on the other hand allocates a probability to the parameters using $\mathcal{Y}$ and a subjective estimate of its distribution. In this work, we use the Bayesian perspective to calibrate the model, which is aptly termed *Bayesian calibration*. This method is commonly chosen for two reasons: it assigns probabilities to possible outcomes rather than outputting a single value and allows the incorporation of certain subjective knowledge.

To introduce the idea of Bayesian calibration, we revisit the idea of conditional probability from probability theory. For two events A and B with $\mathbb{P}(A) \neq 0, \mathbb{P}(B) \neq 0$, the probability of occurrence of both events $\mathbb{P}(A \cap B)$ is given as,

$$\mathbb{P}(A \cap B) = \mathbb{P}(A \mid B)\mathbb{P}(B) = \mathbb{P}(B \mid A)\mathbb{P}(A),$$

where $\mathbb{P}(B \mid A)$, denotes the conditional probability and is read as: probability of occurrence of B given that A has already occurred. However, if the two events are independent, $\mathbb{P}(A \cap B) = \mathbb{P}(B)\mathbb{P}(A)$. Upon rewriting the above conditional probability,

$$\mathbb{P}(A \mid B) = \frac{\mathbb{P}(B \mid A)\mathbb{P}(A)}{\mathbb{P}(B)} \tag{5.4}$$

we obtain the *Bayes' rule*. Bayesian calibration stems from the Bayes' rule of conditional probability. Before we explain each of the terms, we rewrite (5.4) in terms of the probability densities as follows,

$$\pi(\xi \mid \mathcal{Y}) = \frac{\mathcal{L}(\xi, \mathcal{Y})\pi(\xi)}{\mathbb{E}_\pi(y)}. \tag{5.5}$$

Each of the terms in the expression are defined as follows,

- $\pi(\xi \mid \mathcal{Y})$: **Posterior** is the probability of observing the model parameters given a set of observations.
- $\mathcal{L}(\xi, \mathcal{Y})$: **Likelihood** measures the probability of the observations given a set of model parameters. Naturally, this is computed using the model.
- $\pi(\xi)$: **Prior** incorporates subjective knowledge about the parameter.
- $\mathbb{E}_\pi(y)$: **Evidence** is the normalisation term.

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}.$$

The aim is to compute the posterior $\pi(\xi \mid y)$ that provides input parameters that optimally explain the observations. We begin by incorporating our subjective belief about the parameter distribution by assigning it a prior, $\xi \sim \pi(\xi)$. Using (5.2) and (5.3), the *discrepancy model* $\pi(y \mid \xi)$ is defined as follows,

$$\pi(y \mid \xi) := \mathcal{N}\left(y \mid \mathcal{M}(t, \xi), \boldsymbol{\Sigma}\right). \tag{5.6}$$

Using the N independent measurements, the likelihood function is given as:

$$\begin{aligned} \mathcal{L}(\xi, \mathcal{Y}) &:= \prod_{i=1}^{N} \mathcal{N}(y_i \mid \mathcal{M}(t, \xi), \boldsymbol{\Sigma}), \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{(2\pi)^m \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2} \left(y_i - \mathcal{M}(t, \xi)\right)^\top \boldsymbol{\Sigma}^{-1} \left(y_i - \mathcal{M}(t, \xi)\right) \right). \end{aligned} \tag{5.7}$$

We now substitute the above relation in (5.5) to obtain the solution of the inverse problem:

$$\pi(\boldsymbol{\xi} \mid \boldsymbol{y}) = \frac{1}{\mathbb{E}_{\pi}(\boldsymbol{y})} \pi(\boldsymbol{\xi}) \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i \mid \mathcal{M}(t, \boldsymbol{\xi}), \boldsymbol{\Sigma}). \tag{5.8}$$

Since an analytical solution is not available to solve the above problem for practical applications, we resort to *Markov Chain Monte Carlo* (MCMC) simulations. In this work, the MCMC simulations are carried out using the *Affine Invariant Ensemble Algorithm* (AIES) algorithm in the UQLab Toolbox [35].

Since practically the covariance matrix $\boldsymbol{\Sigma}$ is not known, we use the following simplification $\boldsymbol{\Sigma} = \varsigma^2 \mathbb{I}$, where $\mathbb{I} \in \mathbb{R}^{m \times m}$ is an identity matrix and $\varsigma^2 = \mathbb{V}[\delta_i], i = 1, \ldots, m$. The parameter ς^2 is assigned a prior denoted by $\pi(\varsigma^2)$ and is also updated during the MCMC runs.

Finally, we also compute the distribution of the model output $\boldsymbol{y}$ given the data $\boldsymbol{\mathcal{Y}}$ which is called the *posterior predictive distribution*,

$$\pi(\boldsymbol{y} \mid \boldsymbol{\mathcal{Y}}) = \int_{\mathcal{D}_{\xi}} \pi(\boldsymbol{y} \mid \boldsymbol{\xi}) \pi(\boldsymbol{\xi} \mid \boldsymbol{\mathcal{Y}}) \, \mathrm{d}\boldsymbol{\xi}. \tag{5.9}$$

5.4.1. Calibrating E using the model

In this section we calibrate the Young's Modulus E used in the Chaboche Model, since it serves as a good benchmark problem to understand the relation between sensitivity analysis and model calibration. We are aware that calibrating the Young's Modulus from a physical point of view is a well understood task. However, the purpose of this section is clearly to demonstrate the Bayesian approach for model calibration and investigate the usage of surrogate models in that context.

To calibrate the parameter we must solve equation (5.8) using the MCMC algorithm, which requires model evaluations to evaluate the likelihood (5.7). Since it must search through a large space input space, it typically requires a large number of model evaluations and this is the main bottleneck in the calibration task. Therefore, the goal is to first benchmark the performance of the calibration experiment using the expensive model evaluations and try to replace them with the computationally cheaper surrogates previously introduced.

From the sensitivity studies (refer Figures 5.7, 5.8), we have noted that the Young's modulus has the highest sensitivity in the initial linear region. In [p.4-5 16], it has been argued that variance-based sensitivity analysis can be used to infer parameter identifiability. Therefore we choose the initial linear region to calibrate E. This choice is also in-line with the problem physics, since the parameter E solely governs the behaviour in the linear regime. However in general, inferring the identifiability of a parameter using sensitivity indices is an interesting question and deserves an independent treatment (refer [36], for an overview). For example [p.11 13] suggests that a parameter with Total-order Sobol index (computed using MC sampling) less than an empirical threshold, is non-identifiable.

In order to choose a time point t_1 for the calibration, we must compute the minimum time at which yielding will begin. The priors used for the calibration are the same as the bounds of the sensitivity analysis listed in Table 5.1. Therefore, the first instance of plasticity begins at $t = \frac{\sigma_{y_0}}{E \times \dot{\varepsilon}} = \frac{200}{2.1 \times 10^5 \times 5 \times 10^{-2}} \approx 0.019$s, for the set of parameters with the smallest $\sigma_{y|0} = 200\,\text{MPa}$ and the largest $E = 210,000\,\text{MPa}$. Since the time discretisation is $\Delta t = 1 \times 10^{-3}$, we choose the point $t_1 = 0.01$s for calibration. (Although more points can be chosen in the region, they are not investigated here).

For the purposes of this work, we generate mock data for calibration. The assumed true parameters are the same as in 4.2. However currently, only the value $E = 200,000\,\text{MPa}$ is of interest. The stress is therefore:

$$\hat{\sigma}_{xx} = E \cdot \dot{\varepsilon} \cdot t_1 = 200,000 \cdot 5 \times 10^{-2} \cdot 0.01 = 100\,\text{MPa},$$

and the experimental data can be generated from $y_i \sim \mathcal{N}(\hat{\sigma}_{xx}, 10^{-4})$, where the discrepancy parameter is $\varsigma = 0.1$ (since this is only an elementary calibration of the parameter, a small value is chosen for the variance). We calibrate the model using $N = 10$ values from this distribution. The AIES algorithm is run for $4,000$ steps with 4 parallel chains, which results in $16,000$ model evaluations. The histograms of the chosen priors of the model parameters are plotted in Figure 5.14.

Parameter	Mean	Std	Std (Prior)	True Value
E	$200,000$	17	$2,877.77$	$200,000$
σ_{y_0}	320	42	57.74	250
Q_{iso}	240	160	144.3	50
b_{iso}	190	320	288.7	100
n	2.4	1.2	1.43	2
D	51	31	28.58	100

Table 5.3.: Posterior Marginals after calibration with the AIES algorithm using model evaluations. Standard deviations of prior presented for comparison.

The posterior marginals $\pi(\xi \mid y)$ of the parameters ξ are given in Table 5.3. The standard deviation of the parameter E has reduced from $2,877.77$ to 17, indicating that the parameter has converged. As expected, the standard deviations of other parameters have not changed significantly and are therefore not calibrated. The histograms of the posterior marginals of the parameters are plotted in Figure 5.15. The $16,000$ MCMC runs required $\sim$ 18 minutes. This was relatively fast, since the model was solved only upto $t = 0.01$s. (Currently, the Chaboche solver written in Python is called in UQLab using a wrapper, which makes the process inefficient. The calibration can accelerated by completely shifting to Python.)

The posterior predictive distribution is plotted in Figure 5.15 along with the data used in the calibration. The mean of the posterior predictive is 100.0143 MPa and the variance

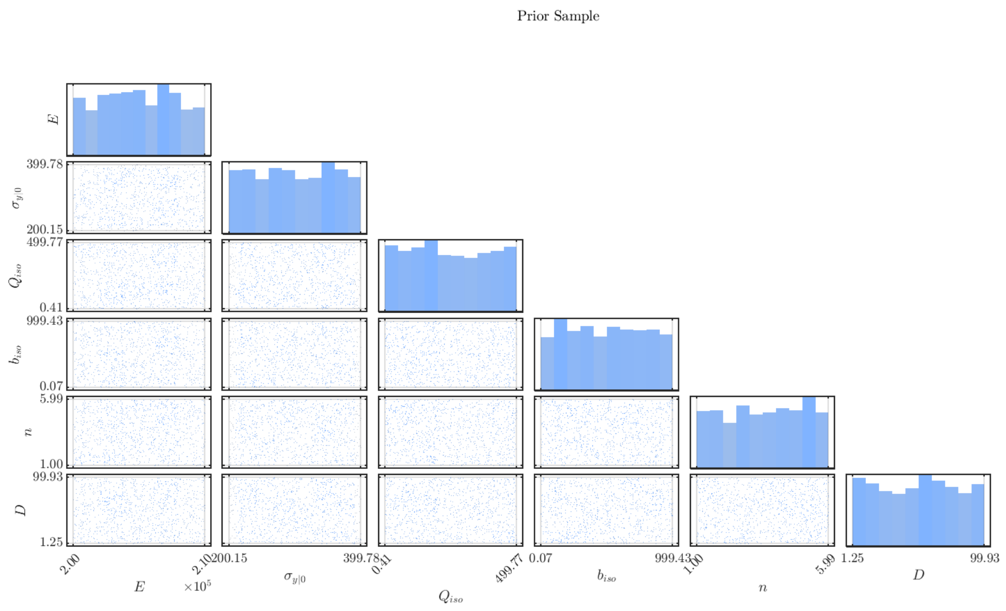


Figure 5.14.: Histograms of the priors of the parameters as in Table 5.1.

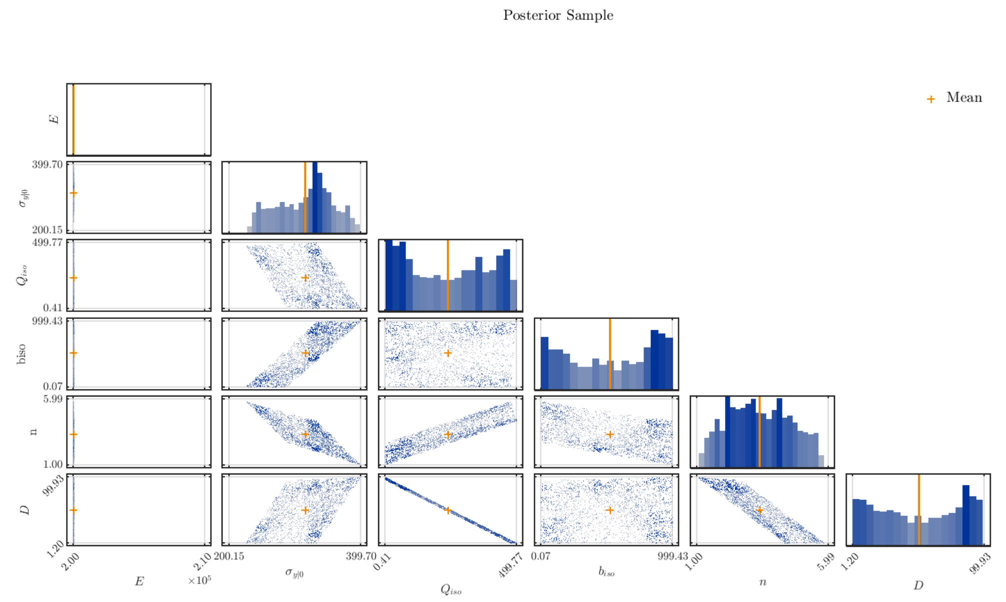


Figure 5.15.: Posterior marginals of the parameters $\pi(\xi \mid y)$ after 16,000 MCMC runs.

is 5.5878×10^{-4} and appears to be in good agreement with the chosen distribution.

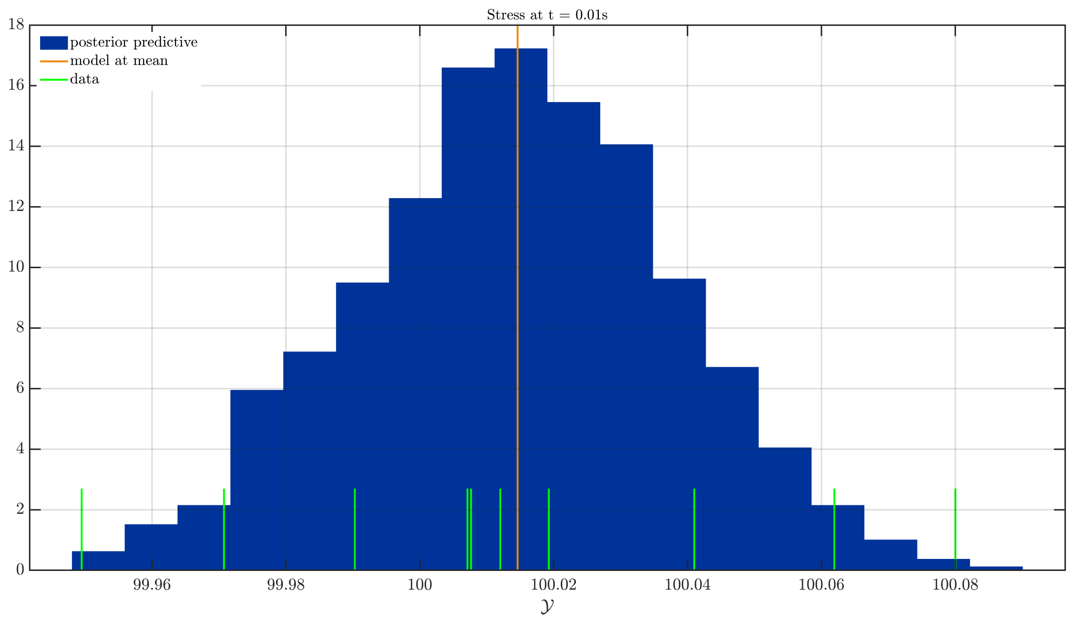


Figure 5.16.: Posterior predictive distribution of the model output σ_{xx}.

5.4.2. Calibrating E using PCE surrogates

In order to accelerate the model calibration, we employ a PCE surrogate to replace the model evaluations necessary in the MCMC runs. Using the same experimental setup from the previous section, we replace the model with a time-frozen surrogate at $t_1 = 0.01$s with a maximum polynomial degree $p = 6$. The marginals of the parameters after $16,000$ MCMC runs are listed in Table 5.4. We note that the PCE surrogate also calibrates the Young's Modulus correctly and the other parameters are not calibrated since their standard deviations do not change.

In the discussion in section 5.1.1, where we built time-frozen PCEs surrogates, we concluded that although time-frozen PCE surrogates generally deteriorate with time, the PCE surrogate for the monotonic loading case seems to deteriorate slowly (refer Figure 5.3) and is therefore able to calibrate E accurately. It is expected that the time-frozen PCE surrogate will be able to replace model evaluations for further time points in the monotonic loading case. However for the cyclic loading case, a similar error analysis must be conducted before replacing the model. (Due to the current workflow, the calibration using the PCE surrogate required $\sim$ 45 mins. This is about 3 times longer than the model evaluations and is counterproductive. The significant decrease in performance is possibly due to the interfacing between Python and MATLAB, because for every surrogate evaluation, the

Parameter	Mean	Std	Std (Prior)	True Value
E	$200,000$	17	$2,877.77$	$200,000$
σ_{y_0}	320	42	57.74	250
Q_{iso}	240	160	144.3	50
b_{iso}	530	320	288.7	100
n	3.4	1.2	1.43	2
D	51	31	28.58	100

Table 5.4.: Posterior Marginals after calibration with AIES using PCE surrogate evaluations. Standard deviations of prior presented for comparison.

surrogate data is first read from a JSON file and the surrogate object is created in Python, which makes it inefficient.)

5.4.3. Calibrating E using KLE+PCE surrogates

To circumvent the shortcomings of time-frozen PCE surrogates in section 2.1.6, we examine KLE+PCE surrogates for model calibration.

Using the same experimental setup from the previous section, we replace the model evaluations with the KLE+PCE surrogate evaluations. The KLE+PCE surrogate is constructed using a truncation order $N_{kl} = 9$ and total polynomial degree $p = 7$ for the gPCE approximation of the modes. The posterior marginals of the parameters after the MCMC runs are listed in Table 5.3. Clearly, the parameter E has not converged and the parameter is not calibrated.

Parameter	Mean	Std	Std (Prior)	True Value
E	$200,000$	$3,000$	$2,877$	$200,000$
σ_{y_0}	320	46	57.74	250
Q_{iso}	290	140	144.3	50
b_{iso}	680	320	288.7	100
n	3.5	1.1	1.43	2
D	42	28	28.58	100

Table 5.5.: Posterior Marginals after calibration with AIES using KLE+PCE surrogate evaluations. Standard deviations of prior presented for comparison.

Whilst evaluating the response of the model for the true parameters, the response of the model and PCE surrogate was 100.0000 MPa, whereas the KLE+PCE surrogate outputs 102.4713. The vast difference in the response and the poor calibration results can be attributed to the poor approximation of the KLE+PCE surrogate in areas of low variance as discussed in section 5.2.2 (refer Figure 5.6).

6. Conclusion and Outlook

6.1. Conclusion

Sensitivity analysis enables us to quantify the contributions of the stochastic inputs of a model towards the output uncertainty. For models with a scalar output, this is typically accomplished using Sobol indices, a variance-based, model-agnostic sensitivity metric. For models with vector-valued outputs, such as discrete time responses, one may be tempted to simply compute the indices for each output. This is however inadequate, since it treats each output in isolation and does not incorporate the time-dependence of the output. In order to provide a holistic and history-aware metric, Alexandrian et al. [3] propose the use of generalised Sobol Indices, which extend the idea of Sobol indices. In this work, we apply this new metric to compute the sensitivities of the time-dependent stress response of the Chaboche model, a constitutive model for viscoplastic materials undergoing cyclic loading.

For most practical problems, a direct computation of the indices using Monte Carlo techniques, is not feasible. This is because, they converge slowly and obtaining a reasonable approximation of the model response manifold requires a large number of expensive model evaluations. To circumvent this problem, surrogate models are typically employed to approximate the manifold which enables us to efficiently extract the desired statistical properties. In this work, we use two different surrogate modeling techniques, generalised Polynomial Chaos Expansion (PCE) and Karnhunen-Loève Expansion with PCE approximated modes (KLE+PCE), to approximate the stochastic response of the Chaboche model. Using the surrogates, we compute both, the Sobol indices and the generalised Sobol indices. We discuss the practical computation of both surrogate modeling techniques and highlight their strengths and weaknesses.

Finally, using the results of the sensitivity analysis, we carry out an elementary calibration of the Young's modulus in an attempt to understand the implications of parameter sensitivity on parameter identifiability. The parameter is calibrated using Bayesian calibration and the resulting inverse problem is solved using Markov Chain Monte Carlo simulations. Since these simulations require model evaluations, we once again resort to the previously computed surrogates to alleviate the computational expense. Here we encounter a key limitation of the KLE+PCE surrogate in areas of low variance due to which it fails to calibrate the parameter.

6.2. Outlook

Although there are several open questions that demand further investigation, we highlight some avenues to pursue the calibration of the Chaboche model. The calibration of the model has three main tasks: determining settings under which a parameter is identifiable, Bayesian calibration and surrogate modeling to accelerate the MCMC runs. We elaborate these tasks as follows:

- In literature a relation between sensitivity analysis and parameter identifiability has been demonstrated for certain models [36]. The sensitivity analysis carried out in this work provides time points at which parameters have high sensitivity/identifiability under a given strain rate (refer Figure 5.7). This can be repeated for different strain rates under montonic and cyclic loading conditions and response under stress controlled and displacement controlled experiments.
- Before any calibration can be carried out, it is quintessential to build surrogates that accurately approximate the statistical properties of the model response. Since the computational expense for obtaining the response of the model especially under cyclic loading can be a significant hurdle. In this regard, local PCE surrogates or a global KLE+PCE surrogate can be employed after careful consideration of their strengths and weaknesses discussed in this work.
- The Bayesian calibration must incorporate the model response under various settings and time points in its likelihood function. For example, the stress response of the model for various strain rates can be incorporated and the time points at which they are considered can be identified using sensitivity analysis. A recent work on calibration of the Chaboche model using the displacement response can be found in [1].
- Implementation changes: In the current work, the Bayesian calibration was carried out using UQLab in MATLAB and the Chaboche model solver and surrogates were written in Python. Although a coupling between the two was possible using function wrappers, it was observed to be very inefficient. Carrying out all computations in one language would be significantly more efficient. One possible way is to carry out all the computations in Python using PyMC3 [25] or emcee [14] (which using AIES for MCMC) or using UQ[py]Lab (only beta version currently available).

Bibliography

[1] Ehsan Adeli et al. “Comparison of Bayesian methods on parameter identification for a viscoplastic model with damage”. In: *Metals* 10.7 (2020), pp. 1–25. ISSN: 20754701. DOI: 10.3390/met10070876.

[2] Alen Alexanderian. *A brief note on the Karhunen-Loève expansion*. 2015. arXiv: 1509.07526 [math.PR].

[3] Alen Alexanderian, Pierre A. Gremaud, and Ralph C. Smith. “Variance-based sensitivity analysis for time-dependent processes”. In: *Reliability Engineering & System Safety* 196 (2020), p. 106722. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2019.106722. URL: https://www.sciencedirect.com/science/article/pii/S0951832019303837.

[4] Andrzej Ambroziak and Paweł Kłosowski. “The elasto-viscoplastic Chaboche model”. In: *TASK QUARTERLY* 10 (Jan. 2006), pp. 49–61. URL: https://www.researchgate.net/publication/228699800_The_elasto-viscoplastic_Chaboche_model.

[5] Ilias Bilionis and N. Zabaras. “Multi-output local Gaussian process regression: Applications to uncertainty quantification”. In: *Journal of Computational Physics* 231 (2012), pp. 5718–5746.

[6] G Blatman and B Sudret. “Sparse polynomial chaos expansions of vector-valued response quantities”. In: *Safety, Reliability, Risk and Life-Cycle Performance of Structures and Infrastructures*. Ed. by George Deodatis, Bruce Ellingwood, and Dan Frangopol. CRC Press, 2014, pp. 3245–3252. ISBN: 978-1-138-00086-5. DOI: \url{10.1201/b16387-469}.

[7] G. Blatman. “Adaptive sparse polynomial chaos expansions for uncertainty propagation and sensitivity analysis”. PhD thesis. Université Blaise Pascal, Clermont-Ferrand, France, 2009.

[8] Emanuele Borgonovo. “Measuring uncertainty importance: Investigation and comparison of alternative approaches”. In: *Risk Analysis* 26.5 (Oct. 2006), pp. 1349–1361. DOI: 10.1111/J.1539-6924.2006.00806.X.

[9] Emanuele Borgonovo and Elmar Plischke. “Sensitivity analysis: A review of recent advances”. In: *European Journal of Operational Research* 248.3 (2016), pp. 869–887. ISSN: 03772217. DOI: \url{10.1016/j.ejor.2015.06.032}.

[10] Ronaldo I Borja. *Plasticity: Modeling & Computation*. Springer Science & Business Media, 2013. URL: https://link.springer.com/book/10.1007%2F978-3-642-38547-6.

[11] Pierre Brémaud. *Probability Theory and Stochastic Processes*. Springer, 2020. URL: https://link.springer.com/book/10.1007%2F978-3-030-40183-2.

[12] J.L. Chaboche. "Constitutive equations for cyclic plasticity and cyclic viscoplasticity". In: *International Journal of Plasticity* 5.3 (1989), pp. 247–302. ISSN: 0749-6419. DOI: https://doi.org/10.1016/0749-6419(89)90015-6. URL: https://www.sciencedirect.com/science/article/pii/0749641989900156.

[13] Simona Dobre et al. "Limits of variance-based sensitivity analysis for non- identifiability testing in high dimensional dynamic models". In: (2012).

[14] D. Foreman-Mackey et al. "emcee: The MCMC Hammer". In: *PASP* 125 (2013), pp. 306–312. DOI: 10.1086/670067. eprint: 1202.3665.

[15] Fabrice Gamboa et al. *Sensitivity analysis for multidimensional and functional outputs*. 2013. arXiv: 1311.1797 [stat.AP].

[16] Hoshin V. Gupta and Saman Razavi. "Revisiting the Basis of Sensitivity Analysis for Dynamical Earth System Models". In: *Water Resources Research* 54.11 (2018), pp. 8692–8717. DOI: https://doi.org/10.1029/2018WR022668. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018WR022668.

[17] Sven Heinrich. "Mehrskalenmodell zur numerischen Schädigungsanalyse von Stahlstrukturen". PhD thesis. Institut für Statik, 2021.

[18] Jan Kaschube. "Numerische Analyse von Stahlkonstruktionen unter Berücksichtigung thermomechanischer Beanspruchung". MA thesis. Institut für Statik, Dec. 2018.

[19] Olivier Le Maitre and Omar M Knio. *Spectral Methods for Uncertainty Quantification, with Applications to Computational Fluid Dynamics*. Springer Science & Business Media, 2010.

[20] Chu V. Mai and Bruno Sudret. "Surrogate Models for Oscillatory Systems Using Sparse Polynomial Chaos Expansions and Stochastic Time Warping". In: *SIAM/ASA Journal on Uncertainty Quantification* 5.1 (2017), pp. 540–571. DOI: 10.1137/16M1083621. eprint: https://doi.org/10.1137/16M1083621. URL: https://doi.org/10.1137/16M1083621.

[21] Stefano Marelli, Nora Lüthen, and Bruno Sudret. *UQLab user manual – Polynomial Chaos Expansions*. Tech. rep. Report # UQLab-V1.4-104. Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich, Switzerland, 2021.

[22] David A B Miller. "An introduction to functional analysis for science and engineering". In: (). arXiv: 1904.02539.

[23] Ulrich Römer. *Lecture notes in Methods of Uncertainty Analysis and Quantification*. May 2020.

[24] A Saltelli. *Global sensitivity analysis: The primer*. Chichester England and Hoboken NJ: John Wiley, 2008. ISBN: 9780470059975.

[25] John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. “Probabilistic programming in Python using PyMC3”. In: *PeerJ Computer Science* 2 (Apr. 2016), e55. doi: 10.7717/peerj-cs.55. url: https://doi.org/10.7717/peerj-cs.55.

[26] I.M Sobol. “Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates”. In: *Mathematics and Computers in Simulation* 55.1 (2001). The Second IMACS Seminar on Monte Carlo Methods, pp. 271–280. issn: 0378-4754. doi: https://doi.org/10.1016/S0378-4754(00)00270-6. url: https://www.sciencedirect.com/science/article/pii/S0378475400002706.

[27] B Sudret. *Uncertainty propagation using polynomial chaos expansions.* 2016. url: https://www.uni-weimar.de/fileadmin/user/fak/bauing/professuren_institute/grk1462/SummerSchool2016/Sudret-WeimarPCE.pdf.

[28] Bruno Sudret. “Global sensitivity analysis using polynomial chaos expansions”. In: *Reliability Engineering and System Safety* 93.7 (2008), pp. 964–979. issn: 09518320. doi: 10.1016/j.ress.2007.04.002.

[29] Bruno Sudret. “Polynomial chaos expansions and stochastic finite element methods”. In: Dec. 2014, p. 624. url: https://ethz.ch/content/dam/ethz/special-interest/baug/ibk/risk-safety-and-uncertainty-dam/publications/reports/RSUQ-2015-008.pdf.

[30] Endre Süli and David F Mayers. *An introduction to numerical analysis.* Cambridge university press, 2003. url: http://newdoc.nccu.edu.tw/teasyllabus/111648701013/Numerical_Analysis.pdf.

[31] Timothy John Sullivan. *Introduction to Uncertainty Quantification.* Vol. 63. Springer, 2015.

[32] William F Trench. “Elementary differential equations with boundary value problems”. In: (2013). url: http://ramanujan.math.trinity.edu/wtrench/texts/TRENCH_DIFF_EQNS_I.PDF.

[33] Rohit Tripathy and Ilias Bilionis. “Deep UQ: Learning deep neural network surrogate models for high dimensional uncertainty quantification”. In: *J. Comput. Phys.* 375 (2018), pp. 565–588.

[34] P.-R. Wagner et al. “Bayesian calibration and sensitivity analysis of heat transfer models for fire insulation panels”. In: *Engineering Structures* 205 (Feb. 2020), p. 110063. issn: 0141-0296. doi: 10.1016/j.engstruct.2019.110063. url: http://dx.doi.org/10.1016/j.engstruct.2019.110063.

[35] P.-R. Wagner et al. *UQLab user manual – Bayesian inversion for model calibration and validation.* Tech. rep. Report # UQLab-V1.4-113. Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich, Switzerland, 2021.

[36] Xu Wu, Koroush Shirvan, and Tomasz Kozlowski. “Demonstration of the relationship between sensitivity and identifiability for inverse uncertainty quantification”. In: *Journal of Computational Physics* 396 (2019), pp. 12–30. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2019.06.032. URL: https://www.sciencedirect.com/science/article/pii/S0021999119304401.

[37] Dongbin Xiu. *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton university press, 2010.

[38] Dongbin Xiu and George Em Karniadakis. “The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations”. In: *SIAM Journal on Scientific Computing* 24.2 (2002), pp. 619–644. DOI: 10.1137/S1064827501387826. eprint: https://doi.org/10.1137/S1064827501387826. URL: https://doi.org/10.1137/S1064827501387826.

[39] Louie L. Yaw. “Nonlinear Static - 1D Plasticity - Various Forms of Isotropic Hardening”. Jan. 2012. URL: https://gab.wallawalla.edu/~louie.yaw/plasticitypublications/1Dplasticity.pdf.

A. Chaboche Model Solver

This appendix is intended to compliment the author's Python implementation and serve an informal documentation for the code in addition to the comments within the code.

For the Newton-Raphson iterations in the implicit scheme, we require the Jacobian $\boldsymbol{J}_{S_{EQ}}$ to be computed. For the 1D Chaboche model, the Jacobian $\boldsymbol{J}_{S_{EQ}} \in \mathbb{R}^{3\times 3}$. The Jacobian can be computed using a finite difference(FD) scheme and some root finding algorithms even compute the jacobian using FD when it is not explicitly supplied. This is however computationally expensive and therefore it is more efficient to provide the closed form of the jacobian which just requires function evaluations. In this case, the finite difference implementation can be used to verify if the derived jacobian is correct.

$$\boldsymbol{J}_{S_{EQ}}(\boldsymbol{Z}_i) := \begin{bmatrix} \frac{\partial F}{\partial \sigma_{xx,i+1}} & \frac{\partial F}{\partial K_{i+1}} & \frac{\partial F}{\partial X_{xx,i+1}} \\ \frac{\partial g}{\partial \sigma_{xx,i+1}} & \frac{\partial g}{\partial K_{i+1}} & \frac{\partial g}{\partial X_{xx,i+1}} \\ \frac{\partial H}{\partial \sigma_{xx,i+1}} & \frac{\partial H}{\partial K_{i+1}} & \frac{\partial H}{\partial X_{xx,i+1}} \end{bmatrix} = \begin{bmatrix} J_1 & J_2 & J_3 \\ J_4 & J_5 & J_6 \\ J_7 & J_8 & J_9 \end{bmatrix}_{[3\times 3]} \tag{A.1}$$

Computing the elements of the Jacobian

We first begin by grouping common terms in the equations [similar to 18], which allows us to present the somewhat tedious derivation in an efficient format. This is also a computationally efficient way to implement the solver since the common terms are evaluated only once thereby avoiding multiple function calls to compute the same expressions.

There are four terms f_b, f_c, f_d, f_e listed below appear multiple times in the system of equations and the jacobian and therefore it is best to compute them at the beginning of the iteration.

$$f_b\left(\sigma_{i+1}\right) := E^{-1} \cdot \left(\frac{\sigma_{xx,i+1} - \sigma_{xx,i}}{\Delta t}\right), \quad f_c\left(\sigma_{i+1}, X_{i+1}\right) := \frac{1}{\sigma_{v,i+1}}$$

$$f_d\left(\sigma_{i+1}, K_{i+1}, X_{i+1}\right) := \left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n, \quad f_e\left(\sigma_{i+1}, X_{i+1}\right) = \sigma_{xx,i+1} - X_{xx,i+1}$$

We recollect some of the definitions from the main text here for convenience,

$$\sigma_v = \sqrt{(\sigma_{xx} - X_{xx})^2} = |\sigma_{xx} - X_{xx}|$$

$$\sigma_{ex} := \sigma_v - (\sigma_{y|0} + K)$$

Common derivatives

The following terms are common derivatives that occur multiple times in the jacobian elements and therefore it is optimal to compute them at the beginning of the iteration.

$$\frac{\partial \sigma_v}{\partial \sigma_{xx}} = \frac{1}{\sigma_v}(\sigma_{xx} - X_{xx}) = \frac{\sigma_{xx} - X_{xx}}{|\sigma_{xx} - X_{xx}|} = sign(\sigma_{xx} - X_{xx}) = f_c \cdot (\sigma_{xx} - X_{xx})$$

$$\frac{\partial \sigma_v}{\partial X_{xx}} = -\frac{\partial \sigma_v}{\partial \sigma_{xx}}, \quad \frac{\partial \sigma_{ex}}{\partial \sigma_{xx}} = \frac{\partial \sigma_v}{\partial \sigma_{xx}}$$

$$\frac{\partial f_d}{\partial \sigma_{ex}} = n \cdot \left\langle \frac{\sigma_{ex}}{D} \right\rangle^{(n-1)} \cdot \frac{1}{D}$$

Common terms in row 1 of the Jacobian.

$$\frac{\partial f_b}{\partial \sigma_{xx,i+1}} = \frac{1}{\Delta t} \cdot E^{-1}$$

$$\frac{\partial f_c}{\partial \sigma_{xx,i+1}} = \left(-\frac{1}{\sigma_{v,i+1}^2}\right) \cdot \frac{\partial \sigma_{v,i+1}}{\partial \sigma_{xx,i+1}} = -f_c^2 \cdot \frac{\partial \sigma_{v,i+1}}{\partial \sigma_{xx,i+1}}$$

$$\frac{\partial f_d}{\partial \sigma_{xx,i+1}} = \frac{\partial f_d}{\partial \sigma_{ex,i+1}} \frac{\partial \sigma_{ex,i+1}}{\partial \sigma_{xx,i+1}}$$

$$\frac{\partial f_e}{\partial \sigma_{xx}} = 1$$

$$\frac{\partial \sigma_{ex,i+1}}{\partial K_{i+1}} = -1$$

Common terms in row 2 of the Jacobian.

$$\frac{\partial f_d}{\partial K_{i+1}} = \frac{\partial f_d}{\partial \sigma_{ex,i+1}} \cdot \frac{\partial \sigma_{ex,i+1}}{\partial K_{i+1}}$$

Common terms in row 3 of the Jacobian.

$$\frac{\partial f_c}{\partial X_{xx,i+1}} = -f_c^2 \cdot \frac{\partial \sigma_{v,i+1}}{\partial X_{xx,i+1}}$$

$$\frac{\partial f_d}{\partial X_{xx,i+1}} = \frac{\partial f_d}{\partial \sigma_{ex,i+1}} \frac{\partial \sigma_{ex,i+1}}{\partial X_{xx,i+1}}$$

$$\frac{\partial f_e}{\partial X_{xx}} = -1$$

Jacobian Row 1

$$F = \varepsilon_{i+1} - \varepsilon_i - \Delta t \cdot \left\{ \underbrace{E^{-1} \cdot \left(\frac{\sigma_{xx,i+1} - \sigma_{xx,i}}{\Delta t}\right)}_{f_b(\sigma_{i+1})} + \underbrace{\frac{1}{\sigma_{v,i+1}}}_{f_c(\sigma_{i+1}, X_{i+1})} \cdot \underbrace{\left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n}_{f_d(\sigma_{i+1}, K_{i+1}, X_{i+1})} \cdot \dot{\varepsilon}_0 \cdot \underbrace{(\sigma_{xx,i+1} - X_{xx,i+1})}_{f_e(\sigma_{i+1}, X_{i+1})} \right\}$$

$$J_1 = \frac{\partial F}{\partial \sigma_{xx,i+1}} = -\Delta t \cdot \left\{ \frac{\partial f_b}{\partial \sigma_{xx,i+1}} + f_c \cdot f_d \cdot \frac{\partial f_e}{\partial \sigma_{xx,i+1}} \cdot \dot{\varepsilon}_0 + \frac{\partial f_c}{\partial \sigma_{xx,i+1}} \cdot f_d \cdot f_e \cdot \dot{\varepsilon}_0 + f_c \cdot \frac{\partial f_d}{\partial \sigma_{xx,i+1}} \cdot f_e \cdot \dot{\varepsilon}_0 \right\}$$

$$J_2 = \frac{\partial F}{\partial K_{i+1}} = -\Delta t \cdot \left\{ f_c \cdot \frac{\partial f_d}{\partial K_{i+1}} \cdot f_e \cdot \dot{\varepsilon}_0 \right\}$$

$$J_3 = \frac{\partial F}{\partial X_{xx,i+1}} = -\Delta t \cdot \left\{ f_c \cdot f_d \cdot \frac{\partial f_e}{\partial X_{xx,i+1}} \cdot \dot{\varepsilon}_0 + f_d \cdot f_e \cdot \frac{\partial f_c}{\partial X_{xx,i+1}} \cdot \dot{\varepsilon}_0 + f_c \cdot f_e \cdot \frac{\partial f_d}{\partial X_{xx,i+1}} \cdot \dot{\varepsilon}_0 \right\}$$

Jacobian Row 2

$$g = K_{i+1} - K_i - \Delta t \cdot b_{iso} \cdot (Q_{iso} - K_{i+1}) \cdot \underbrace{\left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n}_{f_d(\sigma_{i+1}, K_{i+1}, X_{i+1})}$$

$$J_4 = \frac{\partial g}{\partial \sigma_{xx,i+1}} = -\Delta t \cdot b_{iso} \cdot (Q_{iso} - K_{i+1}) \cdot \frac{\partial f_d}{\partial \sigma_{xx,i+1}}$$

$$J_5 = \frac{\partial g}{\partial K_{i+1}} = 1 - \Delta t \cdot b_{iso} \cdot \left\{ (-1) \cdot f_d + (Q_{iso} - K_{i+1}) \cdot \frac{\partial f_d}{\partial K_{i+1}} \right\}$$

$$J_6 = \frac{\partial g}{\partial X_{xx,i+1}} = -\Delta t \cdot b_{iso} \cdot (Q_{iso} - K_{i+1}) \cdot \frac{\partial f_d}{\partial X_{xx,i+1}}$$

Jacobian Row 3

$$H = X_{xx,i+1} - X_{xx,i} - \Delta t \cdot b_{kin} \Big(\frac{2}{3} Q_{kin} \cdot \underbrace{\frac{1}{\sigma_{v,i+1}}}_{f_c(\sigma_{i+1}, X_{i+1})} \underbrace{(\sigma_{xx,i+1} - X_{xx,i+1})}_{f_e(\sigma_{i+1}, X_{i+1})} - X_{xx,i+1} \Big) \cdot \underbrace{\left\langle \frac{\sigma_{ex,i+1}}{D} \right\rangle^n}_{f_d(\sigma_{i+1}, K_{i+1}, X_{i+1})}$$

$$J_7 = \frac{\partial H}{\partial \sigma_{xx,i+1}} =$$

$$-\Delta t \cdot b_{kin} \cdot \left[\left(\frac{2}{3} Q_{kin} \cdot f_c \cdot f_e - X_{xx,i+1} \right) \cdot \frac{\partial f_d}{\partial \sigma_{xx,i+1}} + \frac{2}{3} Q_{kin} \left(f_e \cdot \frac{\partial f_c}{\partial \sigma_{xx,i+1}} + f_c \cdot \frac{\partial f_e}{\partial \sigma_{xx,i+1}} \right) \cdot f_d \right]$$

$$J_8 = \frac{\partial H}{\partial K_{i+1}} = -\Delta t \cdot b_{kin} \cdot \left(\frac{2}{3} Q_{kin} \cdot f_c \cdot f_e - X_{i+1} \right) \cdot \frac{\partial f_d}{\partial K_{i+1}}$$

$$J_9 = \frac{\partial H}{\partial X_{xx,i+1}} = 1 - \Delta t \cdot b_{kin} \cdot \left[\left(\frac{2}{3} Q_{kin} \cdot f_c \cdot f_e - X_{xx,i+1} \right) \cdot \frac{\partial f_d}{\partial X_{xx,i+1}} + \right.$$

$$\left. \left(\frac{2}{3} Q_{kin} \cdot f_e \cdot \frac{\partial f_c}{\partial X_{xx,i+1}} + \frac{2}{3} Q_{kin} \cdot f_c \cdot \frac{\partial f_e}{\partial X_{xx,i+1}} - 1 \right) \cdot f_d \right]$$

B. Chaboche Model Sensitivity Analysis Results

This section presents the unconverged results of the sensitivity analysis of the Chaboche model. Figure B.1 presents the results of the Pointwise-in-time Sobol indices of the Chaboche model under monotonic loading using $10,000$ MC samples. The analysis required $N(N_p + 2) = 10,000(6+2) = 80,000$ model evaluations.

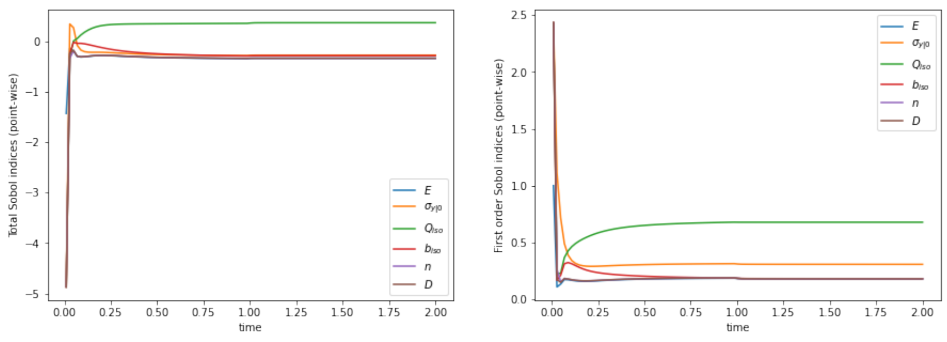


Figure B.1.: **Unconverged** Pointwise-in-time Sobol indices of the Chaboche model under monotonic loading.

In order to alleviate the cost of model evaluations, we replace the model with the KLE+PCE surrogate in the Pick and Freeze estimator. As seen in Figures B.2, B.3. The KLE+PCE surrogate performs poorly (indices are not in the range [0,1]) in the area of low variance.

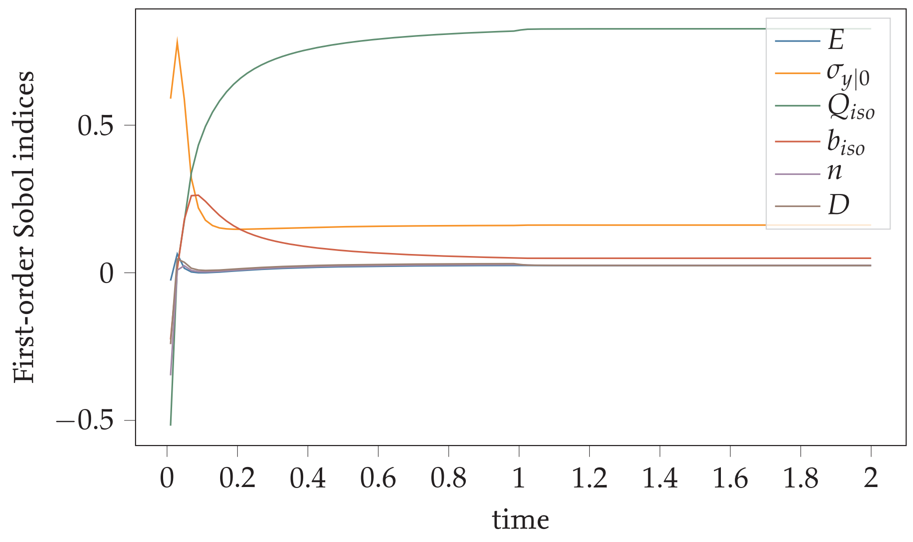


Figure B.2.: Evolution of the First-order Sobol indices of the Chaboche model computed using a KLE+PCE surrogate.

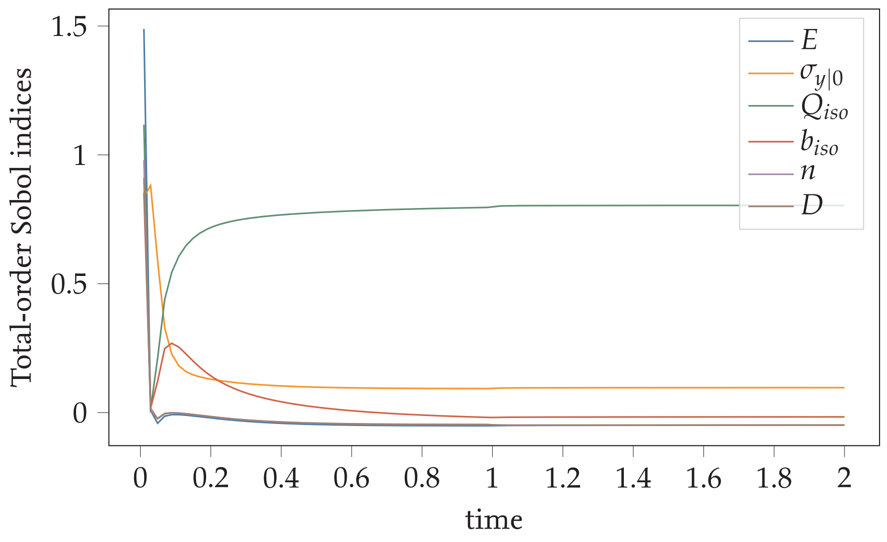


Figure B.3.: Evolution of the Total-order Sobol indices of the Chaboche model computed using a KLE+PCE surrogate.

C. Legendre Polynomials

Example 2.1.1.2 Consider the monomials $\{1, \xi, \xi^2\}$, where $\xi_i \sim \mathcal{U}(-1,1)$. We can apply Gram-Schmidt orthogonalisation to the monomial w.r.t the PDF $f_{\xi_i}(\xi) = \frac{1}{2}$ (obtained using equation 1.13) as follows,

$$\hat{P}_n(\xi) = \xi^n - \textstyle\sum_{j=1}^{n-1} P_j(\xi) \frac{P_j \cdot \xi^n}{P_j \cdot P_j}$$
$$P_n(\xi) = \hat{P}_n(\xi) / \hat{P}_n(1)$$

The monomials 1 and ξ are orthogonal since, $< 1, \xi >= \int_{-1}^{1} 1 \cdot \xi \cdot \frac{1}{2}\, \mathrm{d}\xi = 0$. Similarly ξ and ξ^2 are orthogonal, i.e. $< \xi, \xi^2 >= 0$. We can orthogonalise the third term ξ^2 as follows,

$$\hat{P}_2(x) = \xi^2 - 1 \cdot \frac{< 1, \xi^2 >}{< 1, 1 >} - \underbrace{\xi \cdot \frac{< \xi, \xi^2 >}{< \xi, \xi >}}_{=0}$$

$$\hat{P}_2(x) = \xi^2 - 1 \cdot \frac{< 1, \xi^2 >}{< 1, 1 >} = \xi^2 - 1 \cdot \frac{\int_{-1}^{1} 1 \cdot \xi^2 \cdot \frac{1}{2}\, \mathrm{d}\xi}{\int_{-1}^{1} 1 \cdot 1 \cdot \frac{1}{2}\, \mathrm{d}\xi} = \xi^2 - 1 \cdot \frac{2/3 \cdot 1/2}{1}$$

$$\hat{P}_2(x) = \xi^2 - \frac{1}{3} = \frac{1}{3}(3\xi^2 - 1)$$

We now scale the polynomial as per convention,

$$P_2(x) = \frac{\xi^2 - \frac{1}{3}}{(1^2 - 1\frac{1}{3})} = \frac{1}{2}(3\xi^2 - 1)$$