Quadrature Methods for Wind Turbine Load Calculations

van den Bos, Laurent

DOI
10.4233/uuid:0ed85902-051f-49a9-a99d-dad082fea758

Publication date
2020

Document Version
Final published version

Citation (APA)
van den Bos, L. (2020). *Quadrature Methods for Wind Turbine Load Calculations*. [Dissertation (TU Delft), Delft University of Technology]. Delft University of Technology. https://doi.org/10.4233/uuid:0ed85902-051f-49a9-a99d-dad082fea758

Quadrature Methods for Wind Turbine Load Calculations

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. T. H. J. J. van der Hagen,
chair of the Board for Doctorates,
to be defended publicly on
Tuesday 4 February 2020 at 12:30 o'clock

by

Laurentius Martinus Maria VAN DEN BOS

Master of Science in Industrial and Applied Mathematics,
Eindhoven University of Technology, the Netherlands,
born in Heerlen, the Netherlands.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus	Chairperson
Prof. dr. G. J. W. van Bussel	Delft University of Technology, promotor
Dr. ir. W. A. A. M. Bierbooms	Delft University of Technology, copromotor
Dr. ir. B. Sanderse	Centrum Wiskunde & Informatica, copromotor

Independent members:

Prof. dr. ir. C. W. Oosterlee	Delft University of Technology
Prof. dr. D. T. Crommelin	University of Amsterdam
Prof. dr. M. Muskulus	Norwegian University of Science and Technology
Dr. A. Narayan	University of Utah, United States of America
Prof. dr. S. J. Watson	Delft University of Technology, reserve member

Paranimphs: Prashant Kumar
Esmée Stelten-van den Bos

Printed by: Ipskamp Printing B.V.

Front & back cover: Photo of a wind turbine (by Norbert Aepli, Switzerland) depicted using nodes with various sizes and colors, which are generated using Floyd–Steinberg dithering with small additive noise.

The research for this thesis was carried out in the Scientific Computing (SC) group of the Centrum Wiskunde & Informatica, the Dutch national research institute for mathematics and computer science. Most of the work was conducted in close cooperation with the Section Wind Energy of the faculty of Aerospace Engineering (AE) at Delft University of Technology. This research is part of the Dutch EUROS program (*Excellence in Uncertainty Reduction of Offshore wind Systems*) with project number 14186, which is (partly) financed by the Dutch Research Council (NWO).

ISBN 978-94-6384-101-6

The most precious thing in life is its uncertainty.

Kenkō, from Tsurezuregusa (Essays in Idleness)

Contents

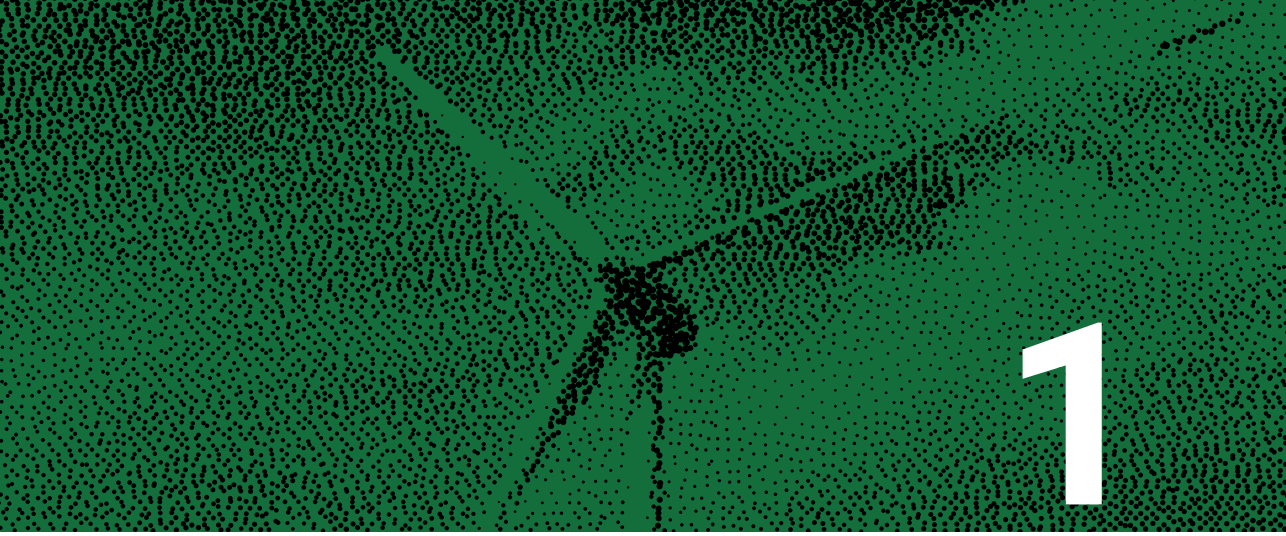
Preface

It is difficult to describe the amount of fun I had over approximately the last four years of doing research. Doing research on uncertainty quantification and wind energy allowed me to come up with some nice algorithms to solve practical problems and, much more importantly, it allowed me to create some nice colorful pictures. I am quite proud of the end result, which is the thesis you have just opened.

From a more practical point of view, the introduction (which starts immediately after this preface) contains a reading guide that outlines the structure of this thesis. The main topic of this thesis is introduced extensively there, so I refrain from doing that here. Many of the remaining chapters are based on existing scientific publications which are, at the time of writing, under review or already published. The details are mentioned in a footnote at the start of each chapter. All references you find in this thesis have been gathered in one single chapter at the end of this thesis, see page 197. Moreover an index has been added (see page 215) that serves as a glossary, a list of nomenclature, and a list of abbreviations. If you stumble upon a term or abbreviation that confuses you, the index will tell you where it has been introduced and is being used elsewhere. If the term is not in the index, you can probably safely ignore its exact definition and continue reading.

With those practicalities out of the way, it is time to dive into the beautiful mathematics of uncertainty quantification and wind energy. Whether you like it or not, I would love to hear from you, so get in touch if you happen to have any questions or comments. For now, I hope you enjoy reading this thesis as much as I did writing it.

Laurent van den Bos
Amsterdam, September 2019



1 Introduction

Offshore wind farms are considered to be an essential part of the transition to renewable energy. Many coastal countries in Europe and elsewhere in the world are constructing wind farms on sea consisting of large numbers of wind turbines [72].

Before an expensive wind farm is constructed in the rapidly varying environment of the rough sea, its construction, design, and maintenance are thoroughly assessed to determine (and to a certain extent maximize) the energy output of the wind farm and the lifetime of the wind turbines. These values can be used to determine the total energy yield over the lifetime of the wind farm and result in a quantity known as the *cost of energy*, which safeguards the financial feasibility of the wind farm.

The procedures to determine the energy output and life time of a wind farm are standardized in the IEC standard [86]. This standard describes various situations that should be taken into account when planning the construction of a wind farm, known as *load cases*. These cases vary from regular power production cases, describing situations in which wind farms produce energy in regular weather conditions, to extreme load scenarios, describing conditions that can potentially severely damage the turbines and significantly affect the power production if not properly considered beforehand.

To ensure that the predictions of the life time of the wind farm are accurate and meaningful, the uncertain nature of the environment, such as weather conditions, must be taken into account. In the IEC standard parameters are described that have to be modeled as inherently uncertain. Examples of such parameters are the mean wind speed, the wind direction, and the wave height. The variation of these parameters makes that the forces acting on the turbines also vary, which affects the lifetime and the cost of energy if not properly accounted for. The randomness of these parameters often comes from external sources, such as the uncertain environmental conditions, and can therefore not be reduced.

A different type of uncertainty arises from the unknown error (or bias) in the model describing the wind farm. This model error follows from the basic fact that the model is imperfect, as there are always modeling assumptions and simplifications that yield a (possibly small) discrepancy between the real value and the modeled value. As the

exact, real value is unknown, the error made by the model can also be interpreted as an uncertainty. However, this uncertainty is of a different type than that arising from the external conditions: it can be reduced by increasing the fidelity of the model, by incorporating more physics, or by simply using a more advanced model. This often comes at a cost: a better model must be derived, implemented, and tested or more computational time is necessary to evaluate the model due to the increased fidelity.

The field of mathematics and engineering that studies these type of problems is often called *uncertainty quantification*. It is common (and prescribed in the IEC standard) to incorporate the uncertainties *non-intrusively*, which means that the variability in the wind turbine performance is assessed by consecutive evaluations of the model. Often this is done using only general characteristics of the model, i.e. the model is treated as a *black box*. The main advantage of a non-intrusive approach is that it does not require any modifications of the model describing the wind turbines.

A possible non-intrusive technique to model the two types of uncertainty is to draw samples and evaluate the computational model for each sample. This approach is commonly known as Monte Carlo sampling. It requires knowledge about the statistical behavior of the uncertain parameters to construct the samples in the first place. The external conditions are often known explicitly; for example, the statistics of the weather conditions are known upfront. Statistically modeling the model error is less straightforward, but by using expert knowledge or by calibrating the model using measurement data, samples can be computed that represent the uncertainty in the model. When both sources of uncertainty are combined into a single probability distribution, large numbers of samples can be drawn from which predictions of the wind turbine performance can be inferred.

However, estimates based on straightforward sampling often converge prohibitively slow. In other words, a large number of samples is necessary to accurately predict the relevant characteristics of the wind turbine. Moreover, obtaining an estimate of the model error requires model evaluations. The computer simulations that model a wind turbine are computationally costly, hence the number of possible model evaluations, and thereby also the number of samples that can be used, is limited. One possible technique to alleviate this is collocation, which consists of deterministically choosing the samples. Hence the samples (often called nodes in a collocation setting) are not drawn from a distribution but chosen using a deterministic algorithm. In a stochastic framework this is often called *stochastic collocation* and this methodology is the key focus of this thesis.

1.1. Stochastic collocation

Stochastic collocation generally describes a family of sampling approaches that treat a stochastic sampling problem in a deterministic way. Various approaches to determine the nodes have been proposed over the years, each with their own advantages and disadvantages. The properties of the model, such as its smoothness properties, are often leveraged to significantly improve the convergence rates of the methods (so fewer model evaluations are necessary for an equally accurate estimate).

Many commonly used collocation methods are inefficient or even inapplicable if considered in the setting of wind turbine calculations. They are based on the assumption

that the underlying input parameters are independently distributed, fully known, and easy to assess statistically. However, these assumptions do not hold for the problem of inferring the statistics of forces on a wind turbine as sketched here. Firstly, the weather conditions are not quantified as independent parameters, for example the turbulence intensity of wind depends on the mean wind speed. Secondly, often these weather conditions do not form well-known distributions, but are only known by a (possibly large) number of measurements. Finally, the model error depends on the model itself, which is non-trivial to assess statistically without requiring a large number of costly model evaluations.

The main goal of this thesis is to construct collocation techniques for the cases in which the underlying model is computationally expensive and the parametric uncertainty must be (partially) inferred from data, such as the case of the problem sketched so far. The model is treated as a black box, so generally no specific knowledge about the underlying equations of the model are incorporated. Only general characteristics of the model, such as whether it responds smoothly to variations in the input, are used. Therefore the approaches are applicable to a wide variety of problems, which are not necessarily related to wind energy.

More specifically, three objectives are addressed in this thesis. The first objective is to derive an efficient stochastic collocation method for the purpose of *uncertainty propagation*, i.e. the probability distribution of the parameters is known a priori (either explicitly or by measurements). The second objective is to apply this method to wind energy load calculations. Notice that these objectives are connected: the stochastic collocation method must be applicable to load calculations. Finally, the third objective is to derive an efficient stochastic collocation method for the case where the probability distribution of the parameters is not known a priori. In particular, calibration problems are considered, where the distribution of the parameters depends on the computationally expensive model.

1.2. Approach of this thesis

The approach taken in this thesis is to construct a polynomial approximation of the model by using a finite number of evaluations and using the polynomial for further post-processing. We will see that the post-processing step is crucial for the construction of the polynomial. If the interest is solely in determining statistical moments of the uncertain forces acting on the turbine, which is often the case, there is a large degree of freedom in the exact location of the nodes. Techniques that leverage this freedom are applicable to a vast class of problems and do not impose severe assumptions on the model under consideration. On the other hand, if the interest is in constructing an exact polynomial which fully approximates the model, there are severe restrictions on the nodes. Nonetheless, globally accurate polynomial approximations might yield more accurate results, depending on the exact quantity of interest. Therefore, the first step of this thesis is to review uncertainty quantification techniques in order to make these type of qualifications mathematically well-defined and to describe the statistical framework used in this thesis.

If the interest is solely in statistical moments of the output uncertainty, such as the mean forces acting on the turbine, it is possible to explicitly derive conditions that

should be imposed on the nodal locations. For this purpose, quadrature rule techniques are used, which are based on strong mathematical theory proving their efficiency and accuracy. The conditions imposed on the nodes to ensure accuracy yield a framework that describes addition, replacement, and removal of nodes in a quadrature rule.

The developed framework can directly be used to construct a quadrature rule tailored to the requirements of wind energy load cases. The proposed quadrature rule, called the *implicit quadrature rule*, only requires samples (instead of a distribution) for its construction. Moreover it yields accurate estimations of statistical quantities for a broad class of computational models.

Based on the developed theory, the focus is shifted to the uncertainty arising from parametric model imperfection. In this case the problem becomes more subtle, as the distribution of the uncertain parameter depends on the model and therefore cannot be straightforwardly assessed. Two methodologies are presented to address this. The first method is based on the construction of an accurate surrogate, designed such that it is accurate in regions where the computational model is close to measurement data. This method can be applied to problems of many types, but shows its true strength if the underlying model is smooth. The second approach that is proposed reuses the implicit quadrature rule. This approach imposes fewer assumptions on the statistical model that describes the relation between the model and the measurement data. Since it is based on a quadrature rule, it is very suitable to infer predictions of the quantity of interest that directly incorporate model uncertainty. However, it cannot be used straightforwardly to construct a surrogate of the computational model under consideration.

Numerical examples are discussed throughout this thesis to demonstrate the effectiveness of each proposed algorithm. For example, the Genz test functions are employed to assess the accuracy of the approaches and the flow over an airfoil (which is especially relevant for wind turbine blade design) is considered to demonstrate the performance of the approaches in complex problems. Moreover, a wind turbine test case is considered to demonstrate the applicability of the algorithms to a standardized load case. For this purpose, the loads acting on a benchmark offshore wind turbine are determined numerically using offshore environmental measurements obtained in the North Sea. As measurement data of the loads of offshore wind farms is not readily available, this case is restricted to the propagation of uncertainty.

1.3. Outline of this thesis

The outline of this thesis follows from the approach sketched above and is depicted in Figure 1.1 in the form of a reading guide. Chapter 2 reviews existing uncertainty quantification techniques and briefly considers their usage in wind energy applications. The focus is mainly on the mathematical foundation that is necessary to tackle the challenges stated here.

In Chapter 3 quadrature rules are studied in detail with a focus on how univariate quadrature rules can be constructed effectively. In this chapter model uncertainty is omitted. By using the derived techniques, the focus shifts to the direct construction of quadrature rules in Chapter 4. Here, the implicit quadrature rule is introduced, which is a quadrature rule that incorporates multiple, correlated parametric uncertainties that are solely known by samples in a single uncertainty propagation framework.

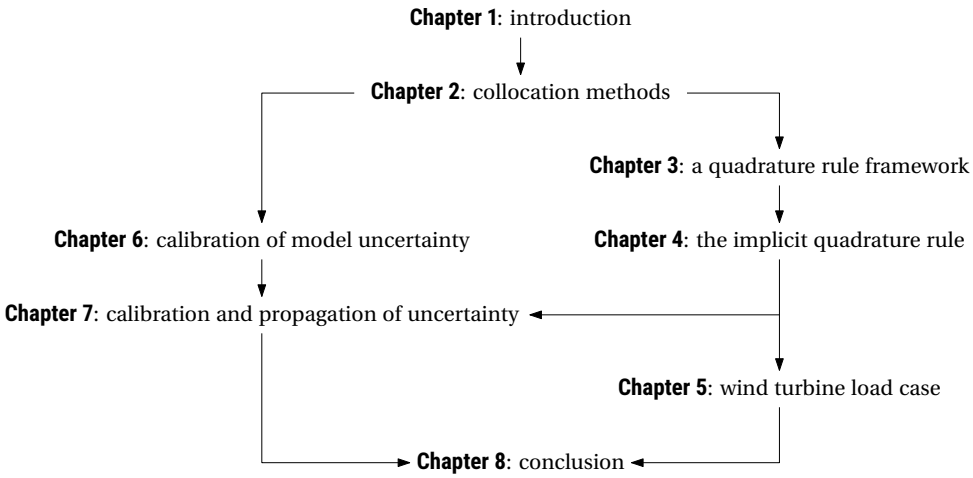
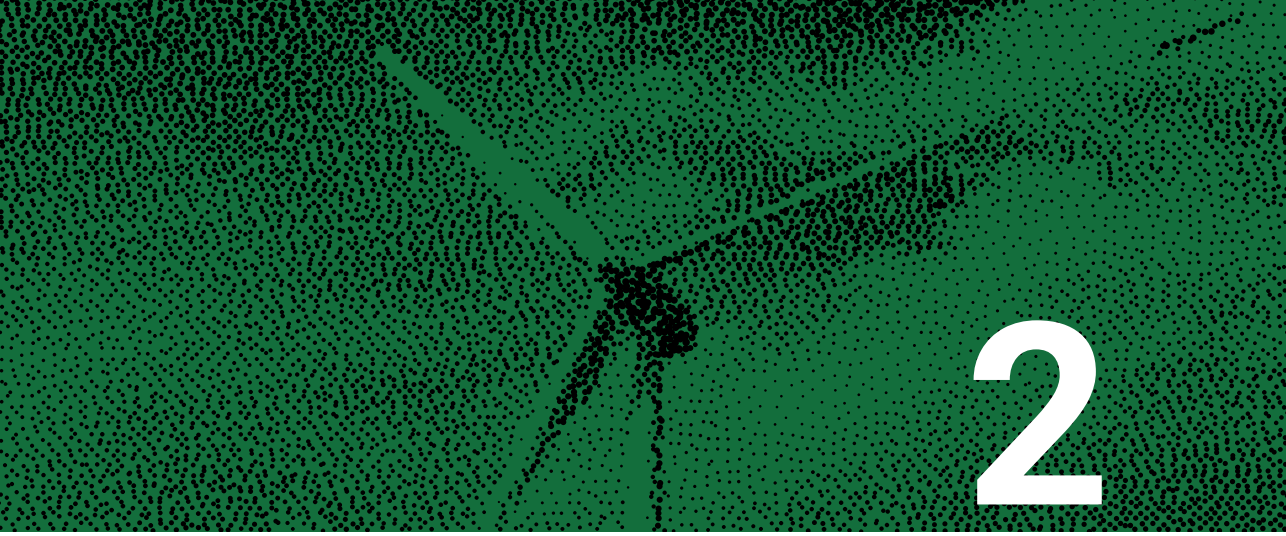


Figure 1.1: Graphical outline of this thesis, where the arrows indicate dependencies between chapters.

The implicit quadrature rule derived in Chapter 4 can readily be used to assess the equivalent loads acting on a wind turbine. Therefore, in Chapter 5 a wind turbine benchmark test case is discussed. For this purpose, the standardized outputs that are commonly used in the wind energy community to assess the performance of a wind turbine are embedded in the probabilistic framework used in this thesis.

These chapters do not incorporate (parametric) model error and therefore in Chapter 6 this type of uncertainty is considered, including how this type can be used in conjunction with a computationally expensive model. For this purpose an interpolant of the expensive model is constructed that is adaptively tailored to the statistical model under consideration. Subsequently, Chapter 7 is concerned with inferring predictions of the model such that model uncertainty is incorporated. For this purpose the quadrature rule framework is extended by exploiting the freedom in the nodal locations as proposed in Chapter 4.

Finally, in Chapter 8 the thesis is summarized and concluded. Moreover some suggestions for further research are discussed.



2 Uncertainty quantification in wind energy: preliminaries

Uncertainties are omnipresent in wind energy applications and quantifying uncertainty in computational models forms the main topic of this thesis. The field of mathematics and engineering that studies such problems is often called *uncertainty quantification*. The goal in that field is to study how uncertainties in parameters and computational models affect the uncertainty in quantities of interest [70, 105, 129, 174, 201]. In this chapter the mathematical background that is relevant for this thesis is introduced and existing uncertainty quantification techniques and principles that are used throughout this thesis are discussed.

The central problem studied in this thesis can be summarized as propagating uncertain parameters, denoted by a multivariate random variable $\mathbf{X}$, through a model, denoted by u. Consequently a new random variable $u(\mathbf{X})$ is obtained. The random variable $\mathbf{X}$ might be known a priori, but problems where this is not the case (e.g. calibration problems) are also considered. To introduce this construction formally, let $(\mathcal{U}, \mathcal{F}, P)$ be a probability space, where $\mathcal{U}$ denotes the set of all outcomes, $\mathcal{F} \subset 2^{\mathcal{U}}$ denotes the accompanying σ-algebra containing all events, and P denotes the probability measure. In the context of uncertainty quantification, it is common to let P denote the uniform distribution, but this is mathematically speaking not necessary. The random parameters are modeled by a d-variate random variable defined in this space, i.e. $\mathbf{X}\colon \mathcal{U} \to \Omega$ with $\Omega \subset \mathbb{R}^d$, where d denotes the number of parameters considered. In this thesis only finite dimensional real-valued random variables are considered. The goal is to infer statistics such as the probability density function or expectation of $u(\mathbf{X})$, where $u\colon \Omega \to \mathbb{R}^n$ is a map that describes a certain computational model.

Throughout this thesis, the underlying random space $(\mathcal{U}, \mathcal{F}, P)$ is omitted from the notation. The computational model under consideration is denoted by u and $\mathbf{X} = [X_1, \ldots, X_d]^{\mathrm{T}}$ are the uncertain input parameters. Moreover it is assumed that all input parameters have a piecewise continuous probability density function, denoted by $\rho(\mathbf{X}) = \rho(X_1, \ldots, X_d)$. Then the statistical properties of $u(\mathbf{X}) = u(X_1, \ldots, X_d)$ can be

assessed without explicit knowledge about the probability space $(\mathcal{U}, \mathcal{F}, P)$. For example, determining the mean of $u(\mathbf{X})$ translates into determining a weighted integral over Ω:

$$\mathbb{E}[u(\mathbf{X})] = \int_{\Omega} u(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{2.1}$$

The variance and other higher order moments can be obtained by replacing $u(\mathbf{X})$ by $u(\mathbf{X})^j$ (for a certain $j = 1, 2, \ldots$).

Globally there are two types of methods to infer statistics about $u(\mathbf{X})$: *intrusive* and *non-intrusive* methods. Intrusive methods consist of expanding the uncertain parameter in a series of known distributions [70] and replacing this expansion in the governing model. The effect of the uncertainty is studied by assessing the obtained model, which requires non-trivial changes to codes built to assess the original model (e.g. codes that numerically solve a partial differential equation). On the other hand, non-intrusive methods rely on existing models and make predictions by repeatedly evaluating the model for various values of the parameters. The focus of this thesis is on the latter, since such methods are most widely applied in the wind energy community. Moreover often the complex computational model describing the wind turbine under consideration cannot be straightforwardly modified to incorporate the uncertain variable. An example where this is the case is considered in Chapter 5.

A key challenge of assessing $u(X_1, \ldots, X_d)$, where $X_1, \ldots, X_d$ are random variables, is that evaluating $u(\mathbf{x})$ for a specific value of $\mathbf{x} \in \Omega$ is computationally costly. Moreover it is assumed that no analytical expression of u is known. For example, u is the solution of a system of partial differential equations or u describes the loads determined by a computer code that models a wind turbine.

If ρ is known explicitly beforehand, this problem is often called *uncertainty propagation*. Many distributions used in wind energy applications fall into this case, e.g. the shape of the distribution describing the uncertain environment is prescribed [85, 86]. This problem is introduced mathematically in Section 2.1, where the approaches that are used in this thesis are outlined.

On the other hand, if ρ is not known a priori, it can be obtained by calibrating the model using measurement data. It is convenient to consider a Bayesian framework for this purpose, as that describes a distribution for the input parameters. This problem is known as *Bayesian model calibration* [95] and encompasses an inverse problem. Predictions under uncertainty, known as *Bayesian predictions*, are among others obtained by propagating the obtained distribution through the model. Calibration allows to infer predictions that incorporate, besides parametric uncertainty, the uncertainty in the model to a certain extent. This uncertainty is often the result from model assumptions, numerical errors, or physical simplifications. Bayesian model calibration is further introduced and discussed in Section 2.2, including its non-trivial computational challenges that are tackled in this thesis.

These concepts are illustrated in Figure 2.1. The core of all methods consists of a model accompanied with uncertain parameters, describing a quantity of interest. The quantity of interest is depicted as $u(\mathbf{X})$ and the interest is in the statistical properties of this random variable (such as the mean). A Bayesian approach consists of formulating a statistical relation between known measurements and a quantity of interest and exploiting this statistical relation to describe the distribution of the input parameters. Bayesian

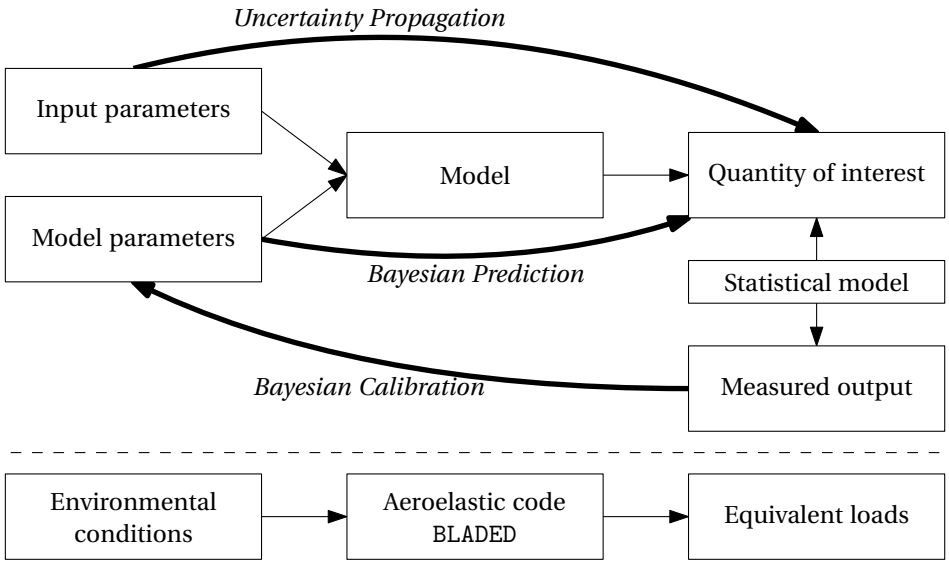


Figure 2.1: Schematic overview of the concepts considered in this thesis and their relation to wind energy load calculations.

prediction consists consequently of propagating these calibrated parameters through the model. The concepts of Bayesian prediction and uncertainty propagation are closely related, but subtly differ due to the usage of measurement data in the Bayesian methods considered in this thesis. The embedding of wind turbine load calculations in the framework (which is further considered in Chapter 5) is illustrated at the bottom of the figure.

2.1. Uncertainty propagation

If the input distribution ρ of $\mathbf{X}$ is known exactly, its properties can be leveraged to accurately assess the statistical properties of $u(\mathbf{X})$. Arguably the best-known methods for this purpose are Monte Carlo methods, where large numbers of samples of $\mathbf{X}$ are used. These methods are widely used in computational sciences due to their simplicity and straightforward error analysis. These methods are briefly discussed in Section 2.1.1.

However, often the number of samples that is necessary to accurately approximate the statistical properties of $u(\mathbf{X})$ is prohibitively large. This can be alleviated by replacing the model u with a surrogate, i.e. an approximation of the model which is computationally cheap to evaluate. In this thesis, the focus is in particular on polynomial interpolation, which is discussed in Section 2.1.2. The accuracy of an interpolating polynomial is highly sensitive to the choice of interpolation nodes. However, if the polynomial is solely used to calculate weighted integrals such as (2.1), the nodal set can be interpreted as a quadrature rule, which allows for significantly more flexibility in the nodal locations. A large part of this thesis is devoted to the construction of quadrature rules and the construction of such sets is discussed in Section 2.1.3.

The goal of this section is not to provide an exhaustive review about all existing propagation methods and there exist many methods for uncertainty propagation that are not mentioned in this thesis. A well-known example is Gaussian process regression (or "Kriging") [152], where a Gaussian process is fitted using evaluations of the computational model. Other methods include using Wiener–Haar expansions [104], employing deep neural networks [181], or clustering approaches [54]. The focus in this thesis is on polynomials due to their strong balance between high accuracy, flexibility, and representability using a computer.

2.1.1. Monte Carlo methods

The idea of Monte Carlo is drawing independently and identically distributed (i.i.d.) samples from $\mathbf{X}$ and evaluating u for each sample. The obtained evaluations form samples of the random variable $u(\mathbf{X})$ and can be post-processed as such, for example

$$\mathbb{E}[u(\mathbf{X})] \approx \frac{1}{K}\sum_{k=0}^{K-1} u(\mathbf{x}_k), \tag{2.2}$$

where $\mathbf{x}_0, \ldots, \mathbf{x}_{K-1}$ are K i.i.d. samples of the random variable $\mathbf{X}$. The accuracy of this method is guaranteed by the central limit theorem, which describes that the mean as estimated by Monte Carlo converges almost surely to the exact mean, or more formally [31]:

$$\sqrt{K}\left(\frac{1}{K}\sum_{k=0}^{K-1} u(\mathbf{x}_k) - \mathbb{E}[u(\mathbf{X})]\right) \xrightarrow{d} \mathcal{N}\left(0, \sigma[u(\mathbf{X})]^2\right).$$

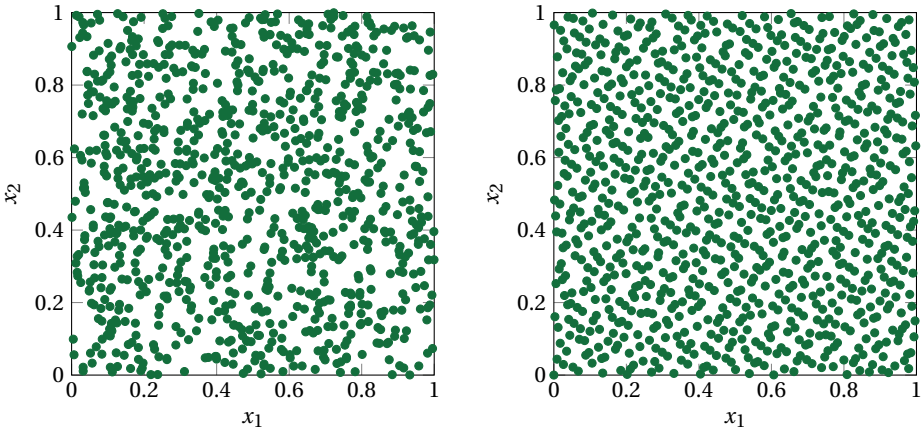
Here, the convergence is in distribution and $\sigma[u(\mathbf{X})]$ denotes the standard deviation of $u(\mathbf{X})$:

$$\sigma[u(\mathbf{X})] = \sqrt{\mathbb{E}\left[(u(\mathbf{X}) - \mathbb{E}[u(\mathbf{X})])^2\right]}.$$

The central limit theorem implies that the estimate from (2.2) converges (approximately) with rate $1/2$, independently of the number of random parameters (the dimension of Ω) and the distribution of the parameters. Often the rate of convergence of the Monte Carlo method is prohibitively slow for the problem at hand, as doubling the accuracy of the estimation requires quadrupling the number of samples. If obtaining an evaluation of the model u is computationally non-trivial, the number of samples necessary to reliably obtain an accurate estimate is simply too large. Notice that this is often the case for wind turbine models.

Monte Carlo methods have been used in wind energy related applications mainly to assess the effect of dependent input parameters, since for mutually independent Gaussian distributed input parameters it is common to use the so-called root-sum-square method [101]. For example, the Monte Carlo method has been employed to study the non-linear effect of correlated uncertainties on the energy yield of an offshore wind farm [57, 100].

Several approaches have been proposed over the years to increase the performance of Monte Carlo methods, either by increasing the convergence rate or by reducing the variance of $u(\mathbf{X})$ (or both). An example of an approach with generally a higher convergence rate is *quasi-Monte Carlo sampling*, where the random i.i.d. samples are



(a) Random samples

(b) Halton set

Figure 2.2: Random samples used for Monte Carlo (*left*) and a deterministic Halton set (*right*) used for quasi-Monte Carlo approaches. Both sets consists of 1000 samples with x_1 and x_2 i.i.d. uniformly distributed on the interval $[0,1]$.

replaced by deterministically determined samples [31], e.g. Halton sequences [183] or Sobol sequences [25]. If the samples distribute well across Ω, an approximately linearly converging estimate is obtained. An example of such a sequence is depicted in Figure 2.2, where the equal distribution across the space $\Omega = [0,1]^2$ can be observed well.

A completely different approach is to incorporate various models with varying accuracy in (2.2) by using the fact that more accurate models are often more computationally expensive. These models can be based on different physics, but can also be obtained by using finer discretization meshes, which are often present if the model represents the numerical solution of a system of partial differential equations. In the latter case it is common to refer to multiple *levels* instead of multiple models. These levels can be exploited by evaluating a large number of samples using a less accurate, but fast mesh and only a small number of samples on the more accurate, but slower mesh. This approach is called *Multilevel Monte Carlo* [71, 99]. If the number of samples that is evaluated on each grid is selected carefully, a significant improvement over straightforward Monte Carlo approaches can be obtained.

2.1.2. Polynomial expansion methods

The idea of non-intrusive polynomial expansions is to replace the computationally costly model with a significantly cheaper approximation, i.e. a polynomial in this thesis. This cheaper approximation is called a *surrogate* or a *response surface*. It is constructed by considering a (possibly multivariate) polynomial basis $\{\varphi_0, \ldots, \varphi_N\} \subset C^\infty(\Omega)$ spanning the vector space $\Phi_N \subset C^\infty(\Omega)$ and subsequently by constructing a $u_N \in \Phi_N$ such that $u_N \approx u$ using a finite number of evaluations of u. Here, Φ_N contains all linear

combinations of $\varphi_0, \ldots, \varphi_N$, i.e.

$$\Phi_N := \operatorname{span}\{\varphi_0, \ldots, \varphi_N\} = \left\{ \sum_{k=0}^{N} a_k \varphi_k \mid a_0, \ldots, a_N \in \mathbb{R} \right\}.$$

The parameter N is a free parameter. Choosing a larger N increases the dimension of Φ_N, resulting into a more accurate u_N. However, we will see that a larger N also requires more computational effort to compute the surrogate. There exist many approaches to construct u_N and two are briefly discussed here due to their relevance for this thesis: *pseudo-spectral approaches* and *interpolation approaches*.

The pseudo-spectral approach [70, 105, 191, 200, 201], or *polynomial chaos expansion*, is based on projecting u onto the space Φ_N. If $\varphi_0, \ldots, \varphi_N$ form an orthonormal basis, the projection u_N can be explicitly denoted by

$$u_N(\mathbf{x}) = \sum_{k=0}^{N} a_k \varphi_k(\mathbf{x}), \text{ with } a_k = \langle \varphi_k, u \rangle_{L^2_\rho(\Omega)}. \tag{2.3}$$

Here, $\langle \varphi_k, u \rangle_{L^2_\rho(\Omega)}$ denotes the well-known inner product on $L^2_\rho(\Omega)$:

$$\langle f, g \rangle^2_{L^2_\rho(\Omega)} = \int_\Omega f(\mathbf{x})\, g(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x}, \text{ with } f, g \in L^2_\rho(\Omega).$$

Notice that we again omit the measure space $(\mathcal{U}, \mathcal{F}, P)$ from the notation. Since u_N is a spectral projection, it holds that [180, 188]

$$u_N = \operatorname*{argmin}_{\varphi \in \Phi_N} \|\varphi - u\|_{L^2_\rho(\Omega)},$$

or equivalently: u_N is the *best approximation polynomial* of u in the space Φ_N measured in the L^2_ρ-norm.

The coefficients a_k of the expansion still depend non-trivially through (2.3) on the model u. Both intrusive [129] and non-intrusive [202] methods can be considered in this regard. Approximating a_k non-intrusively makes it a numerical integration problem and introduces, besides the error of using a finite number of basis vectors in (2.3), a second error term in the expansion, which is why it is often called a *pseudo-spectral expansion*. The coefficients can, for example, be approximated by Monte Carlo methods or more efficiently by means of quadrature rule approaches (which are further discussed in Section 2.1.3).

A different approach is to define the coefficients a_k by means of interpolation. In this case, a_k are defined as the solution of the following system of linear equations:

$$u_N(\mathbf{x}_j) = \sum_{k=0}^{N} a_k \varphi_k(\mathbf{x}_j) = u(\mathbf{x}_j), \text{ for } j = 0, \ldots, N, \tag{2.4}$$

where $\mathbf{x}_0, \ldots, \mathbf{x}_N$ are nodes yielding an interpolant u_N that is exact at the values $\mathbf{x}_k$. Notice that interpolation is also a projection method, since interpolating the interpolant leaves it unchanged.

The linear system described by (2.4) might be singular. For univariate $\mathbf{X}$ the system has a unique solution if all nodes are distinct, but for multivariate $\mathbf{X}$ it is less trivial to ensure that the system above admits a unique solution, and a nodal set is called to be *poised* if it does. All nodal sets constructed in this thesis are by construction poised.

By using the Lagrange basis polynomials as basis for Φ_N, an exact expression of the coefficients a_k as used in (2.4) can be deduced. For this purpose, let $\varphi_k = \ell_k^N$ be the Lagrange basis polynomials with the defining property that

$$\ell_k^N \in \Phi_N \text{ such that } \ell_k^N(\mathbf{x}_j) = \delta_{k,j} = \begin{cases} 1 & \text{if } k = j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

Then the interpolation polynomial u_N can be expressed explicitly as follows:

$$u_N(\mathbf{x}) = \sum_{k=0}^{N} u(\mathbf{x}_k)\, \ell_k^N(\mathbf{x}). \tag{2.6}$$

If Φ_N contains solely univariate polynomials up to degree N, i.e. $\varphi_k(x) = x^k$ for $x \in \mathbb{R}$, the Lagrange basis polynomials can be expressed explicitly as follows:

$$\ell_k^N(x) = \prod_{\substack{j=0 \\ j \neq k}}^{N} \frac{x - x_j}{x_k - x_j}. \tag{2.7}$$

Similar expressions can be derived for multivariate interpolation, though the obtained polynomials are significantly less straightforward. More details about multivariate Lagrange interpolation are provided in Chapter 6.

The accuracy of the interpolant is often measured in the L_ρ^∞-norm by means of the *Lebesgue inequality* [26, 78, 84]. To this end, let $\mathcal{L}_N$ be the interpolation operator that interpolates u, i.e. $\mathcal{L}_N u = u_N$. Then for any polynomial $\varphi \in \Phi_N$ and any piecewise continuous u it holds that

$$\begin{aligned} \|u - \mathcal{L}_N u\|_{L_\rho^\infty(\Omega)} &= \|u - \varphi + \varphi - \mathcal{L}_N u\|_{L_\rho^\infty(\Omega)} \\ &= \|u - \varphi + \mathcal{L}_N \varphi - \mathcal{L}_N u\|_{L_\rho^\infty(\Omega)} \\ &\leq \|u - \varphi\|_{L_\rho^\infty(\Omega)} + \|\mathcal{L}_N(u - \varphi)\|_{L_\rho^\infty(\Omega)} \\ &\leq \left(1 + \|\mathcal{L}_N\|_{L_\rho^\infty(\Omega)}\right) \|u - \varphi\|_{L_\rho^\infty(\Omega)}, \end{aligned} \tag{2.8}$$

where due to continuity of u and φ it holds that $\|u - \varphi\|_{L_\rho^\infty(\Omega)} = \sup_{\rho(\mathbf{x})>0} |u(\mathbf{x}) - \varphi(\mathbf{x})|$. If the distribution ρ is clear from the context, this norm will be simply called the ∞-norm and denoted in this thesis by

$$\|\cdot\|_\infty := \|\cdot\|_{L_\rho^\infty(\Omega)}.$$

The L_ρ^∞-norm of $\mathcal{L}_N$ can be further expanded using the Lagrange basis polynomials from (2.5):

$$\|\mathcal{L}_N\|_{L_\rho^\infty(\Omega)} = \sup_{\|f\|_{L_\rho^\infty(\Omega)}=1} \|\mathcal{L}_N f\|_{L_\rho^\infty(\Omega)} = \max_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} |\ell_k^N(\mathbf{x})|.$$

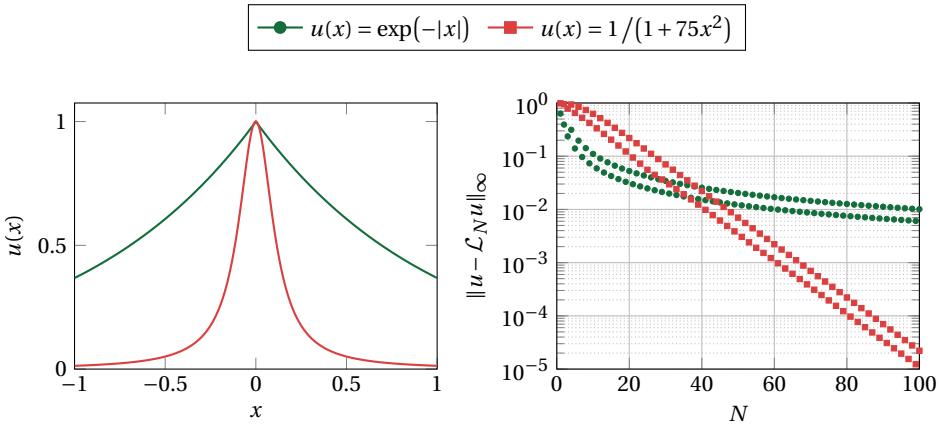


Figure 2.3: Approximation error of a polynomial interpolant of a non-smooth and smooth function. *Left:* Plot of the functions. *Right:* Absolute interpolation error of both functions using Chebyshev extrema, a nodal sequence with logarithmically growing Lebesgue constant.

The Lebesgue inequality is obtained by replacing φ in (2.8) by the best approximation polynomial in Φ_N measured in the L^∞_ρ-norm, obtaining the following:

$$\|u - \mathcal{L}_N u\|_{L^\infty_\rho(\Omega)} \leq (1 + \Lambda_N) \inf_{\varphi \in \Phi_N} \|u - \varphi\|_{L^\infty_\rho(\Omega)}, \text{ with } \Lambda_N = \|\mathcal{L}_N\|_{L^\infty_\rho(\Omega)}. \tag{2.9}$$

The constant Λ_N is known as the Lebesgue constant. It solely depends on the nodes $\mathbf{x}_0, \ldots, \mathbf{x}_N$ and the domain of definition Ω. On the contrary, the best approximation polynomial depends solely on the model u but not on the nodes. This is a major power of the Lebesgue inequality: it splits the interpolation error into a part solely depending on the nodes and a part solely depending on the model. It is one of the most important tools to demonstrate convergence in this thesis.

The absolute error of the best approximation polynomial can be estimated by means of Jackson's inequality [87, 140], which relates it to the modulus of continuity of u. In general, it holds that if u is absolutely continuous and Ω is compact (i.e. Ω is closed and bounded) the best approximation polynomial converges (at least) linearly to the model (for increasing N). If higher order derivatives of u exist and are bounded, the rate of convergence is higher. A large number of results on this topic exists, the interested reader is referred to Watson [188] and the references therein for more information.

Any nodal set has a Lebesgue constant Λ_N that grows at least logarithmically in N [175]. Examples of nodal sets with logarithmically growing Lebesgue constant are Chebyshev nodes of the first kind, Gauss–Lobatto nodes, Fekete nodes, and Chebyshev extrema (the nodes from the Clenshaw–Curtis quadrature rule). Various other types of growth exist, for example Chebyshev nodes of the second kind have a linearly growing Lebesgue constant and equidistant nodes have an exponentially growing Lebesgue constant.

The effect of the type of model on the accuracy of the interpolant is illustrated in Figure 2.3. A smooth function (e.g. $u(x) = 1/\left(1+75x^2\right)$) can be approximated well using polynomials and yields spectral convergence of the interpolation error, provided that the nodal set has a slowly growing Lebesgue constant, whereas non-smooth functions (e.g. $u(x) = \exp\left(-|x|\right)$) in general exhibit algebraic convergence. Nodes with a rapidly growing Lebesgue constant only yield a converging interpolant if the error between the model and its best approximation polynomial (i.e. $\inf_{\varphi\in\Phi_N} \|u-\varphi\|_\infty$) decays very fast, which is typically the case if the model can be globally approximated well using a single Taylor series expansion.

This topic will be further considered in Chapter 6, where interpolation will be used to construct a surrogate tailored to Bayesian calibration problems (which are introduced in Section 2.2).

Many models related to wind energy are computationally expensive, so polynomial approximation has found usage in uncertainty quantification problems with wind energy models. For instance, Petrone et al. [145] uses an interpolating surrogate to study the effect of uncertain meteorological conditions, insect contamination, and manufacturing tolerances. The same authors have also applied this approach to optimization under uncertainty [83] and polynomial surrogates have also been used by various other authors [59, 127, 128, 154]. However, incorporating correlated parameters and tailoring the nodal placement specifically to the problem under consideration remains notoriously difficult.

2.1.3. Quadrature rules

Often the interest is not directly in constructing an accurate surrogate (denoted by u_N), but in approximating integrals over u, for example moments of $u(\mathbf{X})$ such as (2.1) or inner products used in the expansion in (2.3). The straightforward approach is to firstly construct an interpolant u_N of the model u using (2.6) and approximate integrals of u by means of integrals of u_N, obtaining

$$\int_\Omega u(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} \approx \int_\Omega u_N(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_\Omega \sum_{k=0}^{N} \ell_k^N(\mathbf{x})\,u(\mathbf{x}_k)\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} = \sum_{k=0}^{N} u(\mathbf{x}_k) \int_\Omega \ell_k^N(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{2.10}$$

The obtained expression is an *interpolatory quadrature rule*, with nodes $\{\mathbf{x}_k\}_{k=0}^N$ and weights $\{w_k\}_{k=0}^N$ such that

$$\mathbb{E}[u] = \int_\Omega u(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} \approx \mathbb{E}[u_N] = \sum_{k=0}^{N} u(\mathbf{x}_k)\,w_k, \text{ with } w_k = \int_\Omega \ell_k^N(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}.$$

Here, $\mathbb{E}[u(\mathbf{X})]$ is abbreviated to $\mathbb{E}[u]$. In this thesis, the operator that applies the quadrature rule to the model u is denoted by $\mathcal{A}_N$, i.e.

$$\mathcal{A}_N u := \sum_{k=0}^{N} u(\mathbf{x}_k)\,w_k.$$

The accuracy of an interpolatory quadrature rule of this form can be assessed by using the Lebesgue inequality from (2.9), obtaining the following bound:

$$\begin{aligned}|\mathbb{E}[u]-\mathcal{A}_N u| &= |\mathbb{E}[u]-\mathbb{E}[u_N]| \leq \mathbb{E}\big[|u-u_N|\big] = \int_\Omega |u(\mathbf{x})-u_N(\mathbf{x})|\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} \\ &\leq \int_\Omega \|u-u_N\|_{L^\infty_\rho(\Omega)}\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} = \|u-u_N\|_{L^\infty_\rho(\Omega)} \leq (1+\Lambda_N)\inf_{\varphi\in\Phi_N}\|u-\varphi\|_{L^\infty_\rho(\Omega)}. \end{aligned} \tag{2.11}$$

This bound can be sharpened by interpreting a quadrature rule as a weighted sum and releasing the connection with that of the interpolating polynomial. For this purpose, let $\varphi_0,\dots,\varphi_D$ be $D+1$ linearly independent polynomials spanning the space Φ_D such that $\mathcal{A}_N\varphi_j = \mathbb{E}[\varphi_j]$ for all $j=0,\dots,D$. Then $\mathcal{A}_N\varphi = \mathbb{E}[\varphi]$ for all $\varphi\in\Phi_D$. So by directly considering a quadrature rule as a weighted average, it holds for any polynomial $\varphi\in\Phi_D$ that

$$\begin{aligned}|\mathbb{E}[u]-\mathcal{A}_N u| &= |\mathbb{E}[u]-\mathbb{E}[\varphi]+\mathbb{E}[\varphi]-\mathcal{A}_N u| \\ &= |\mathbb{E}[u]-\mathbb{E}[\varphi]+\mathcal{A}_N\varphi-\mathcal{A}_N u| \\ &\leq |\mathbb{E}[u]-\mathbb{E}[\varphi]|+|\mathcal{A}_N\varphi-\mathcal{A}_N u| \\ &\leq \Big(\|\mathbb{E}\|_{L^\infty_\rho(\Omega)}+\|\mathcal{A}_N\|_{L^\infty_\rho(\Omega)}\Big)\|u-\varphi\|_{L^\infty_\rho(\Omega)}.\end{aligned}$$

This derivation is similar to (2.8). The Lebesgue inequality of the quadrature rule operator $\mathcal{A}_N$ can be obtained by noticing that

$$\|\mathcal{A}_N\|_{L^\infty_\rho(\Omega)} = \|\mathcal{A}_N\|_\infty = \sup_{\|f\|_\infty=1}\left|\sum_{k=0}^N f(\mathbf{x}_k)\,w_k\right| = \sum_{k=0}^N |w_k|$$

and

$$\|\mathbb{E}\|_{L^\infty_\rho(\Omega)} = \|\mathbb{E}\|_\infty = \sup_{\|f\|_\infty=1}\left|\int_\Omega f(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}\right| = 1.$$

Hence the following Lebesgue inequality is obtained [24]:

$$|\mathbb{E}[u]-\mathcal{A}_N u| \leq \left(1+\sum_{k=0}^N |w_k|\right)\inf_{\varphi\in\Phi_D}\|u-\varphi\|_{L^\infty_\rho(\Omega)}. \tag{2.12}$$

Here, the sum of the absolute weights is the Lebesgue constant of the quadrature rule operator $\mathcal{A}_N$. If $D=N$, this bound is sharper then the one obtained previously in (2.11):

$$\sum_{k=0}^N |w_k| = \sum_{k=0}^N\left|\int_\Omega \ell_k^N(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}\right| \leq \int_\Omega\sum_{k=0}^N|\ell_k^N(\mathbf{x})|\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} \leq \sup_{\mathbf{x}\in\Omega}\sum_{k=0}^N|\ell_k^N(\mathbf{x})| = \Lambda_N.$$

The key difference that is exploited here is that the weights are integral quantities that average the Lagrange basis polynomials, whereas the Lebesgue constant as defined in (2.9) contains the maximal values of the Lagrange basis polynomials. In particular, if

all weights are non-negative (i.e. $w_k = |w_k|$ for all k), the bound obtained in (2.12) is independent of the exact values of the nodes and the weights:

$$|\mathbb{E}[u] - \mathcal{A}_N u| \leq 2 \inf_{\varphi \in \Phi_D} \|u - \varphi\|_{L^\infty_\rho(\Omega)}.$$

Hence if a quadrature rule firstly integrates all polynomials up to high degree exactly and secondly has positive weights, it provides an accurate means of approximating integrals, provided that the integrand can be approximated using polynomials. So to obtain accurate estimations for a large class of functions it is sufficient to construct a quadrature rule such that the weights are positive and such that the quadrature rule integrates a large number of polynomials.

Similarly, if all weights are non-negative, estimations obtained using a quadrature rule are numerically stable. In particular, if a function u is perturbed by a numerical error $\varepsilon > 0$, say $\tilde{u} = u \pm \varepsilon$, this does not significantly affect $\mathcal{A}_N u$:

$$|\mathcal{A}_N u - \mathcal{A}_N \tilde{u}| \leq \|\mathcal{A}_N\|_\infty \|u - \tilde{u}\|_\infty = \varepsilon.$$

This demonstrates that a quadrature rule with positive weights is numerically stable, regardless of the nodal set under consideration. Based on this principle, it is common to measure the stability of the quadrature rule using its condition number:

$$\kappa_N := \frac{\|\mathcal{A}_N\|_\infty}{\int_\Omega \rho(\mathbf{x})\,\mathrm{d}\mathbf{x}} = \frac{1}{\int_\Omega \rho(\mathbf{x})\,\mathrm{d}\mathbf{x}} \sum_{k=0}^{N} |w_k|. \tag{2.13}$$

This definition facilitates the usage of weighting functions that are not necessarily a probability density function, i.e. weighting functions which have $\int_\Omega \rho(\mathbf{x})\,\mathrm{d}\mathbf{x} \neq 1$.

Since these properties play a central role in this thesis, the following notation is used frequently to denote the exact integrals of the polynomials φ_j:

$$\mu_j = \int_\Omega \varphi_j(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}, \text{ for } j = 0, \ldots, D. \tag{2.14}$$

Unless stated otherwise, it is assumed throughout this thesis that μ_j is exactly known for all j. Notice that in the univariate case these values correspond to the raw moments of the distribution.

To obtain convergence based on the Lebesgue inequality and ensure numerical stable estimates, it is sufficient to ensure that the sum of the absolute values of the weights remains bounded, i.e. $\|\mathcal{A}_N\|_\infty < A$ for a known A that is independent of N. However, a major advantage of having non-negative weights is that it makes the operator $\mathcal{A}_N$ non-negative, so the quadrature rule estimation of a non-negative function is always non-negative. This is is especially relevant for the purposes of uncertainty propagation: the variance is an integral quantity with a non-negative integrand.

The Lebesgue inequality does not describe a sharp upper bound on the error of a quadrature rule. For example, estimating the integral over a discontinuous function by means of a quadrature rule might converge, even though the upper bound of the Lebesgue inequality is a constant. There exist many more convergence results considering quadrature rules, which are not necessarily based on the Lebesgue inequality. The interested reader is referred to Brass and Petras [24] and Brandolini et al. [23].

Univariate quadrature rules

Many quadrature rules with varying properties exist and the search for more efficient rules tailored to specific applications is ongoing [37]. Arguably the best-known univariate quadrature rule is the Gaussian quadrature rule [74], whose nodes are defined as the roots of the N-th orthogonal polynomial in $L^2_\rho(\Omega)$. More specifically, let φ_k be the univariate polynomial of degree k such that for all k and j it holds that

$$\langle \varphi_k, \varphi_j \rangle_{L^2_\rho(\Omega)} = \int_\Omega \varphi_k(x)\,\varphi_j(x)\,\rho(x)\,\mathrm{d}x = \delta_{k,j} = \begin{cases} 1 & \text{if } k = j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.15}$$

Then the nodes of the Gaussian quadrature rule form the roots of these polynomials, i.e. a rule of N nodes is obtained by solving $\varphi_N(x_k) = 0$ for $k = 0, \ldots, N-1$. The idea of computing quadrature rules from the roots of polynomials is explored more extensively in Chapter 3.

The Gaussian quadrature rule has the favorable property that it has positive weights and moreover $N+1$ nodes integrate all polynomials of degree $2N+1$ and less exactly. To see this, let φ be a polynomial of at most degree $2N+1$. Then using polynomial division, this polynomial can be written as follows:

$$\varphi = p\,\varphi_{N+1} + q,$$

where p and q are polynomials of at most degree N. The polynomial p can be written as a linear combination of basis vectors $\varphi_0, \ldots, \varphi_N$, so $p = \sum_k p_k \varphi_k$ with $p_k \in \mathbb{R}$. Hence it holds that

$$\begin{aligned}
\int_\Omega \varphi(x)\,\rho(x)\,\mathrm{d}x &= \int_\Omega \big(p(x)\,\varphi_{N+1}(x) + q(x)\big)\,\rho(x)\,\mathrm{d}x \\
&= \int_\Omega \left(\sum_{k=0}^{N} p_k \varphi_k(x)\right) \varphi_{N+1}(x)\,\rho(x)\,\mathrm{d}x + \int_\Omega q(x)\,\rho(x)\,\mathrm{d}x \\
&= \sum_{k=0}^{N} p_k \int_\Omega \varphi_k(x)\,\varphi_{N+1}(x)\,\rho(x)\,\mathrm{d}x + \int_\Omega q(x)\,\rho(x)\,\mathrm{d}x \\
&= \int_\Omega q(x)\,\rho(x)\,\mathrm{d}x,
\end{aligned}$$

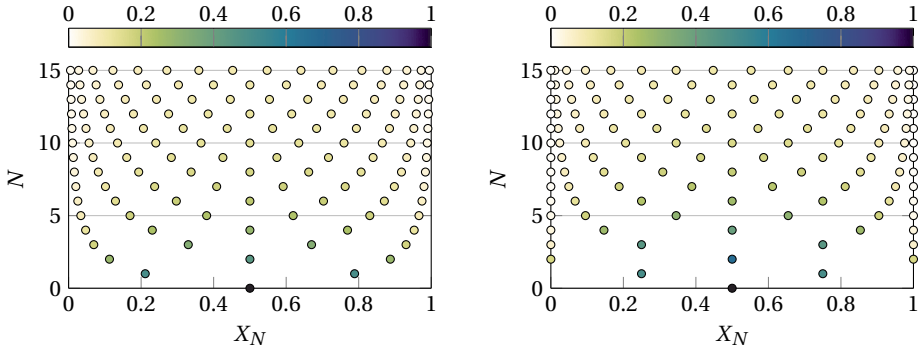
where (2.15) is used, i.e. the orthogonality of the basis vectors. To demonstrate that the Gaussian quadrature rule indeed integrates φ exactly, we use that

$$\varphi(x_k) = p(x_k)\,\varphi_{N+1}(x_k) + q(x_k) = q(x_k).$$

Then

$$\mathcal{A}_N \varphi = \sum_{k=0}^{N} \varphi(x_k)\,w_k = \sum_{k=0}^{N} q(x_k)\,w_k = \int_\Omega q(x)\,\rho(x)\,\mathrm{d}x = \int_\Omega \varphi(x)\,\rho(x)\,\mathrm{d}x.$$

From this, it immediately follows that the weights are all positive. To demonstrate that $w_k \geq 0$ (for any k), let $\varphi = \ell_k^N \cdot \ell_k^N$, where ℓ_k^N is the k-th Lagrange basis polynomial



(a) Gauss–Legendre **(b)** Clenshaw–Curtis

Figure 2.4: The Gauss–Legendre and Clenshaw–Curtis quadrature rules for various numbers of nodes. The colors of the nodes indicate their weight.

from (2.7). Then φ is a polynomial of degree $2N$ with the property that $\varphi(x) \geq 0$ for all x and $\varphi(x_j) = \delta_{k,j}$. Hence

$$\mathcal{A}_N \varphi = \sum_{j=0}^{N} \varphi(x_j)\, w_j = w_k = \int_\Omega \varphi(x)\, \rho(x)\, \mathrm{d}x > 0.$$

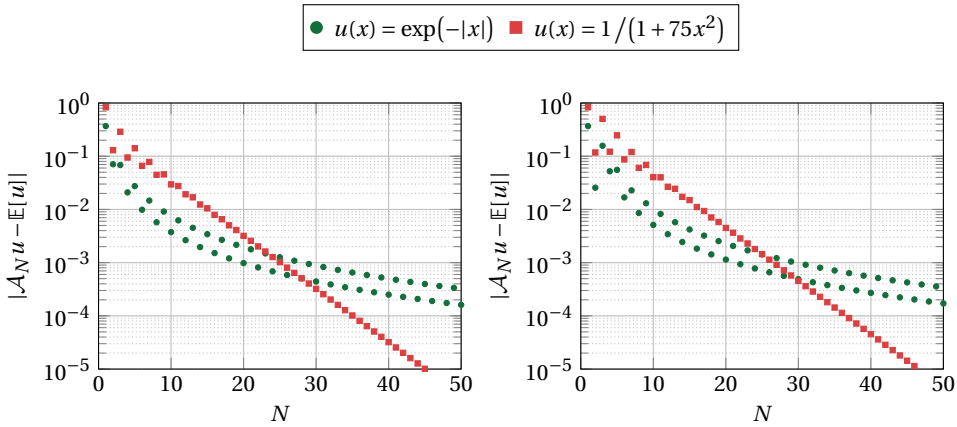
A different commonly used rule is the Clenshaw–Curtis quadrature rule [35], which is formed by the extrema of the Chebyshev polynomials:

$$x_k = \cos\left(\frac{k}{N}\pi\right), \text{ for } k = 0, \ldots, N. \tag{2.16}$$

The Clenshaw–Curtis quadrature rule has positive weights if the uniform distribution is considered and for any other distribution with bounded support the sum of the absolute weights becomes arbitrary close to μ_0 for large N [24]. The quadrature rule is nested for specific levels: it holds that $X_{N_L} \subset X_{N_{L+1}}$ with $N_L = 2^L$ (for $L = 1, 2, \ldots$). Analytic expressions for its nodes and weights are known [180].

The Gauss–Legendre quadrature rule, which is the Gaussian quadrature rule of the uniform distribution, and the Clenshaw–Curtis quadrature rule are depicted in Figure 2.4 for increasing number of nodes. The effect of using these quadrature rules to integrate smooth and non-smooth functions is illustrated in Figure 2.5. Here, the same functions are used as in Figure 2.3. Similarly to interpolation, the smooth function yields spectral convergence and the non-smooth function yields algebraic convergence. The convergence behavior of both quadrature rules that are used is similar, even though the Gaussian quadrature rule has a higher polynomial degree. Their performance is further assessed later in this thesis (see page 63).

A major disadvantage of the Gaussian quadrature rule is that sequences of quadrature rules for increasing N are not nested. Therefore existing evaluations of the integrand



(a) Gauss–Legendre **(b)** Clenshaw–Curtis

Figure 2.5: Absolute integration error of a smooth and non-smooth integrand, using two different interpolatory quadrature rules with positive weights. The domain of integration under consideration is the unit interval and the distribution under consideration is the uniform distribution.

are not reused when refining a quadrature rule estimation by considering a larger quadrature rule. On the other hand, the nodes of the Clenshaw–Curtis quadrature rule are nested, if the number of nodes grows exponentially. However, this rule does not incorporate the distribution of the random variable in its construction. The challenging problem of constructing univariate quadrature rules that are nested, have high degree, and have positive weights is further examined in Chapter 3.

Multivariate quadrature rules

Extending quadrature rules or interpolation nodes to multivariate spaces is not straightforward. Arguably the most naive approach is to use the tensor product of the interpolation or integration operator:

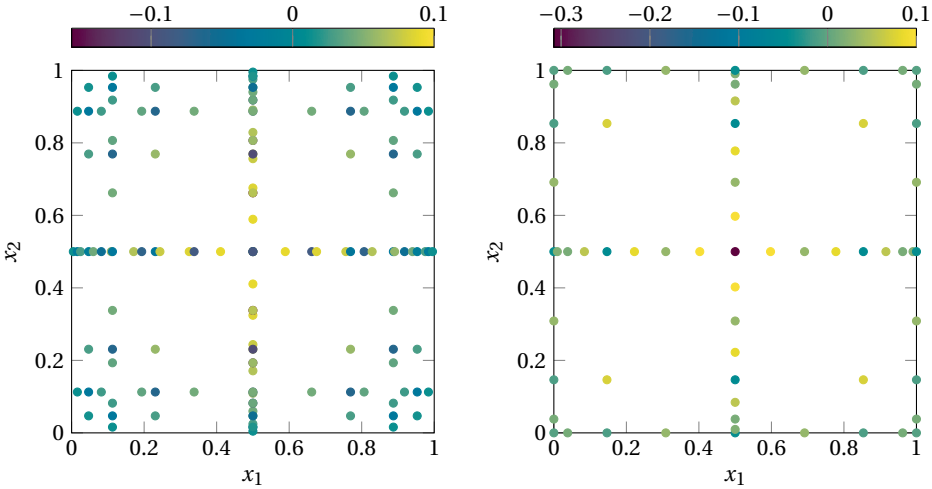
$$\mathcal{A}_{N^d} u = \bigotimes_{i=1}^{d} \mathcal{A}_N^{(i)} u,$$

where $\mathcal{A}_N^{(i)}$ denotes a univariate quadrature rule that is used in the i-th dimension. A similar expression is obtained for interpolation nodes. The tensor grid can only be used to obtain a quadrature rule if the distributions of the random inputs are mutually independent, which implies that $\rho(\mathbf{X})$ can be decomposed as follows:

$$\rho(\mathbf{X}) = \prod_{i=1}^{d} \rho(X_i).$$

In this case, $\mathcal{A}_N^{(i)}$ is a quadrature rule with respect to the distribution $\rho(X_i)$.

The input parameters of wind turbine load calculations are usually not independently distributed. For example, the wind and wave direction are often correlated, and



(a) Gauss–Legendre (142 nodes) **(b)** Clenshaw–Curtis (65 nodes)

Figure 2.6: Two examples of the Smolyak sparse grid constructed using Gauss–Legendre and Clenshaw–Curtis quadrature rules. The colors of the nodes indicate their weight.

should be taken into account as such according to the IEC standard [86]. Therefore it is not straightforward to apply the tensor grid to load calculations.

Notice that the number of nodes of a tensor grid increases exponentially in d: the tensor grid has N^d nodes, which significantly deteriorates the convergence rate of the approach. If r denotes the convergence rate of the univariate quadrature rules, the convergence rate of the quadrature rule estimate of a tensor grid is r/d. This effect is known as the *curse of dimensionality* and demonstrates one of the key differences between Monte Carlo and collocation approaches based on polynomials: Monte Carlo converges often prohibitively slow, but its convergence rate is independent of the dimension, whereas collocation converges very fast for smooth functions, provided that the dimension of Ω is not too large.

An approach to alleviate the curse of dimensionality is the *Smolyak sparse grid* [169]. There are many variants of such a grid, but in this thesis the combination rule is considered [134, 135], which is essentially a multivariate telescopic sum, defined as follows (in operator notation):

$$\mathcal{S}_K = \sum_{\|\mathbf{a}\|_1 \leq K} \bigotimes_{i=1}^{d} \Delta_{a_i}^{(i)}, \text{ with } \mathbf{a} = (a_1, \ldots, a_d),\ \Delta_l^{(i)} = \left(\mathcal{A}_{N_l}^{(i)} - \mathcal{A}_{N_{l-1}}^{(i)}\right), \text{ and } \|\mathbf{a}\|_1 = \sum_{i=1}^{d} a_i. \quad (2.17)$$

Here, N_l denotes the number of nodes used for the l-th level. Often N_l grows linearly or exponentially in l. The number of nodes of a Smolyak sparse grid is greatly reduced if a sequence of nested quadrature rules is used, which is illustrated in Figure 2.6 for a nested Clenshaw–Curtis and a non-nested Gauss–Legendre quadrature rule.

Similar to the tensor grid, the Smolyak sparse grid is only applicable if the distributions of $X_1, \ldots, X_d$ are mutually independent. On the other hand, the convergence rate of this approach is significantly higher compared to the tensor grid. If N_l grows exponentially in l and the univariate quadrature rules used for the construction of the grid are interpolatory and have positive weights, then [92, 201]

$$|\mathbb{E}[u] - \mathcal{S}_K u| \leq A \frac{(\log N)^{(r+1)(d-1)}}{N^r}, \text{ for } u \in C^r(\Omega).$$

Here, N denotes the total number of nodes in the sparse grid and r is the smoothness of u, i.e. u is r times continuously differentiable. The constant A is independent of d and N. The curse of dimensionality is still present, but its effect is significantly alleviated.

The Smolyak sparse grid has the disadvantage that it does not preserve positivity of the weights. Even if a Smolyak sparse grid is constructed using univariate quadrature rules with positive weights, the obtained quadrature rule does generally not have positive weights. Albeit the negative weights, the sparse grid does yield converging estimates under weak assumptions, since the sum of the absolute values of the weights grows logarithmically [135], i.e. $\|\mathcal{S}_K\|_\infty = \mathcal{O}(\log N)$, with a little abuse of notation. However, as stated before, this implies that the estimation of an integral of a positive integrand is not necessarily positive.

The problem of constructing multivariate quadrature rules is further considered in Chapter 4. In that chapter a new quadrature rule with positive weights is proposed, called the *implicit quadrature rule*. It is constructed based on samples (e.g. from a distribution), which makes it particularly applicable to wind turbine load cases. Such an application is considered in Chapter 5.

2.2. Bayesian model calibration

Bayesian model calibration [95] is a systematic approach to calibrate the parameters of a computational model if measurement data of its output is provided. It is based on principles of Bayesian data analysis [65], with the key property that the statistical model contains a (often complex) numerical model. The mathematical formulation describes a probability density function of the parameters, the so-called *posterior*, which can be propagated through the computational model to infer predictions under uncertainty. The calibration of parameters of a model can be interpreted as calibration of the model with respect to measurement data, i.e. the obtained posterior describes in some sense the uncertainty or inherent error of the model and the data.

There exist various alternative ways to calibrate models, describe their uncertainty, or model uncertain model parameters [157]. A popular method is to describe the uncertain model parameters as intervals [55, 56, 109, 176]. In this case no probability framework is used and it is solely assumed that the value of a parameter is encompassed in an (possibly unknown) interval. Interval approaches have been combined with probability, obtaining interval-valued probability measures [189]. This can be further extended to obtain a complete framework for drawing conclusions based on weighted intervals, as described by Dempster–Shafer theory [43, 163]. This framework has been applied successfully to uncertainty propagation [164]. Approaches based on interval arithmetic form viable alternatives to Bayesian model calibration and it is an ongoing discussion

whether probability theory encompasses all alternative uncertainty frameworks that are applied to computational problems [97, 136].

For the purposes of this thesis, the Bayesian framework is employed because it provides a solid and rich mathematical framework to calibrate parameters under uncertainty. Moreover, since it yields a distribution of the parameters, it can be straightforwardly combined with the presented methodologies for uncertainty propagation to obtain a complete framework for quantification and propagation.

2.2.1. Bayes' law

The calibration of models in a Bayesian framework follows naturally from *Bayes' law*. In the most basic setting, let $u\colon \Omega \to \mathbb{R}$ (with $\Omega \subset \mathbb{R}^d$ with $d = 1, 2, \ldots$) be a model depending on certain closure or fitting parameters $\mathbf{x}$. Here, $u(\mathbf{x})$ models a physical phenomenon where $\mathbf{x}$ are parameters that are often introduced to simplify the model or compensate for lack of knowledge. Bayesian model calibration is a systematic approach to infer a probability distribution of $\mathbf{x}$ using measurement data. To this end, let $\mathbf{z} = (z_1, \ldots, z_n)^{\mathrm{T}}$ be a vector with n measurements of the physical phenomenon. The idea is to formulate a statistical relation between the measurement data $\mathbf{z}$ and the model $u(\mathbf{x})$, which is consequently used to infer the parameters $\mathbf{x}$ by means of Bayes' law.

Arguably the most straightforward statistical model is the following:

$$z_k = u(\mathbf{x}) + \varepsilon_k, \text{ for all } k = 1, \ldots, n,$$

where all ε_k are i.i.d. Gaussian random variables with zero mean and known standard deviation σ, i.e. $\varepsilon_k \sim \mathcal{N}(0, \sigma^2)$. Based on this statistical model, a likelihood can be derived, which describes the probability of observing certain data if the parameters $\mathbf{x}$ are known. In this case, it can be derived by noticing that $z_k - u(\mathbf{x}) \sim \mathcal{N}(0, \sigma^2)$. The likelihood, which is denoted by $q(\mathbf{z} \,|\, \mathbf{x})$ throughout this thesis, then readily follows:

$$q(\mathbf{z} \,|\, \mathbf{x}) \propto \exp\left[-\frac{1}{2}\frac{\|u(\mathbf{x}) - \mathbf{z}\|_2^2}{\sigma^2}\right], \text{ with } \|u(\mathbf{x}) - \mathbf{z}\|_2^2 = \sum_{k=1}^{n} (u(\mathbf{x}) - z_k)^2. \tag{2.18}$$

Throughout this thesis, it is assumed that $q(\mathbf{z} \,|\, \mathbf{x})$ is a probability density function, which can be used to determine the scaling factor in this definition.

Prior to calibration, there is often non-trivial knowledge available about $\mathbf{x}$, e.g. upper and lower bounds, results from previous calibration, or expert knowledge. Such knowledge is incorporated in a distribution called the prior, which is denoted by $q(\mathbf{x})$ throughout this thesis. For example, consider $q(\mathbf{x}) = 1$ for $\mathbf{x} \in [-1, 1]^d$ and $q(\mathbf{x}) = 0$ otherwise, describing that the parameters $\mathbf{x}$ are bounded on the unit interval. The prior and the likelihood can be combined into the *posterior*, by application of Bayes' law:

$$q(\mathbf{x} \,|\, \mathbf{z}) \propto q(\mathbf{z} \,|\, \mathbf{x})\, q(\mathbf{x}).$$

Here, the constant of proportionality is called the *evidence*, denoted as follows:

$$q(\mathbf{z}) = \int q(\mathbf{z} \,|\, \mathbf{x})\, q(\mathbf{x})\, \mathrm{d}\mathbf{x}.$$

Hence if the likelihood is described by (2.18) and the prior is uniform as described before, the posterior follows readily:

$$q(\mathbf{x} \mid \mathbf{z}) \propto \begin{cases} \exp\left[-\frac{1}{2}\, \|u(\mathbf{x}) - \mathbf{z}\|_2^2 / \sigma^2\right] & \text{if } \mathbf{x} \in [-1,1]^d, \\ 0 & \text{otherwise.} \end{cases}$$

Ideally, the statistical model should account for all sources of error and avoid underestimation of the uncertainty. For example, the statistical model described above does not account for systematic bias of the computational model. On the other hand, overestimating the errors yields a statistical model whose posterior remains close to the prior, such that the calibration procedure does not meaningfully add a benefit over the prior knowledge. There are various statistical models accounting for various sources of error, e.g. by incorporating model bias [95], by using non-Gaussian likelihoods [206], or by considering a Gaussian process to describe error locality [52, 95].

The usage of Bayesian model calibration for uncertainty assessment is not widespread in the wind energy community, which is possibly related to the high computational cost involved with these procedures (see Section 2.2.2 below). One of the few results is by Van Buren et al. [182], where the focus is specifically on blade design. The high computational cost is alleviated by calibrating a surrogate model instead of the full model. The high computational cost of Bayesian methods is one of the key issues addressed in this thesis.

2.2.2. Markov chain Monte Carlo

As the likelihood explicitly depends on the computational model, the posterior is a well-known distribution only in highly simplified cases. Moreover, it is only known up to a constant, as computing the evidence is often difficult or even impossible. *Markov chain Monte Carlo methods* are Monte Carlo methods specifically tailored to sampling from posterior distributions. The idea of these methods is not to directly compute samples of the posterior, but to construct a sequence of samples $\{\mathbf{x}_k\}_{k=0}^{\infty}$ such that it forms a Markov chain with the posterior as limiting distribution. In this case, a large number of samples forms approximately a sample set from the posterior.

Arguably the best-known Markov chain Monte Carlo method is the *Metropolis–Hastings algorithm* [80, 122]. As the sequence of samples forms a Markov chain, it is only necessary to describe the step between two consecutive samples. Therefore, consider a given sample $\mathbf{x}_k$ and let a sample $\mathbf{x}^*$ be randomly drawn from a proposal distribution that depends on $\mathbf{x}_k$, e.g. a Gaussian distribution with mean equal to $\mathbf{x}_k$. Then $\mathbf{x}_{k+1} = \mathbf{x}^*$ with probability α and $\mathbf{x}_{k+1} = \mathbf{x}_k$ otherwise, with

$$\alpha = \min\left(1, \frac{q(\mathbf{x}^* \mid \mathbf{z})}{q(\mathbf{x}_k \mid \mathbf{z})} \frac{P(\mathbf{x}_k \mid \mathbf{x}^*)}{P(\mathbf{x}^* \mid \mathbf{x}_k)}\right),$$

where P denotes the probability density function of the proposal distribution. An example of samples drawn using this approach is depicted in Figure 2.7, where samples are drawn from the standard Gaussian distribution with a "bad" initial sample $\mathbf{x}_0$ and a Gaussian distributed proposal distribution:

$$\mathbf{x} \mid \mathbf{x}_k \sim \mathcal{N}(\mathbf{x}_k, \sigma^2), \text{ with } \sigma = 1/5.$$

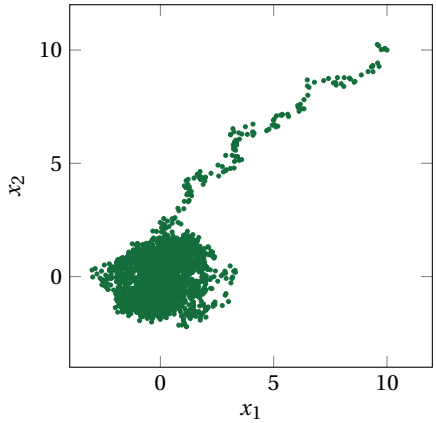


Figure 2.7: 2500 samples drawn from the standard Gaussian distribution using the Metropolis–Hastings algorithm and a Gaussian proposal distribution with standard deviation 1/5. The Markov chain is initiated at $\mathbf{x}_0 = (10, 10)^{\mathrm{T}}$.

The convergence rate of Markov chain Monte Carlo methods is ideally equal to regular Monte Carlo methods (as discussed in Section 2.1.1). However, often a large number of samples is necessary to ensure that the obtained sequence forms samples from the desired distribution P. Therefore it is common to use a *burn-in*, which simply means that a certain number of initially drawn samples is neglected to "forget" the effect of the first samples (notice that a burn-in is clearly visible in Figure 2.7). Moreover, the sequence forms a Markov chain, which implies that subsequent nodes are correlated. It is therefore rarely the case that Markov chain Monte Carlo methods perform as good as regular Monte Carlo methods.

The performance of Markov chain Monte Carlo methods can be improved by using similar approaches as applied to standard Monte Carlo approaches, e.g. by using a surrogate or by replacing the sampling procedure with a quadrature rule. Both approaches are used in this thesis. In Chapter 6 a surrogate is considered, whereas in Chapter 7 a quadrature rule is considered.

2.3. Conclusion and outlook

In this thesis, the problem of uncertainty propagation with a computationally expensive model is considered. Two variants are discussed: one where the distribution is known explicitly (for instance described by a probability density function or by samples) and one where the distribution is calibrated and therefore depends on the model.

If the distribution is known explicitly, there are various existing approaches to construct nodal sets for interpolation or integration. However, many of these approaches require stringent assumptions on the input distribution. Moreover these assumption are not applicable in wind turbine load calculations. In this thesis alternatives are developed, see Chapters 3 and 4. These are applicable to a wide range of problems. Mathematically, the accuracy of the approaches is assessed using test functions. To illustrate the applica-

bility to computationally complex problems, the example of computing the flow over an airfoil incorporating uncertainty (which is a relevant problem for wind turbine blade design) is considered various times in this thesis. Moreover, the approaches are directly applicable to load calculations, which is demonstrated in Chapter 5.

If the distribution is not known explicitly, the problem is more subtle. The usual approach, i.e. Markov chain Monte Carlo, is often not applicable if the model under consideration is computationally complex. Recently alternatives to alleviate this have been developed, which are further extended and generalized in this thesis, see Chapter 6 and 7. Again, the example of calculating the flow over an airfoil is considered.

3

A geometrical interpretation of interpolatory quadrature rules

Quadrature rules are collocation methods that are used oftentimes in this thesis to determine weighted integrals such as (2.1). As discussed in Chapter 2, a quadrature rules converges if it has positive weights and integrates a large number of polynomials, or in other words, has high degree. Moreover, from a computational perspective we are mainly interested in nested quadrature rules, which allow for straightforward refinement of quadrature rule estimations. Many quadrature rules exist that have two of these three properties (i.e. nested, high degree, and positive weights) and it is non-trivial to directly construct rules that have all three properties. The focus of this chapter is not directly on constructing quadrature rules, but on gaining insight and deriving procedures that modify existing quadrature rules, which aid the construction of quadrature rules later in this thesis.

The approach taken is to derive a mathematical framework describing modifications of univariate interpolatory quadrature rules. More specifically, three elementary operations are proposed: the addition of nodes, the removal of nodes, and the replacement of nodes. All operations are designed to preserve positive weights, keep the quadrature rule interpolatory, and by construction yield nested quadrature rules. They form the key ingredients of the proposed methodologies in subsequent chapters.

3.1. Introduction

As discussed briefly in Section 2.1.3, the best-known interpolatory quadrature rule is possibly the Gaussian quadrature rule [74], which exists for virtually any probability

The majority of this chapter is based on the following article: L. M. M. van den Bos and B. Sanderse. A geometric approach for the addition of nodes to an interpolatory quadrature rule with positive weights. *Under review*, 2019. arXiv: 1902.07477 [math.NA].

Section 3.3, that describes the removal of nodes, is based on the following article: L. M. M. van den Bos, B. Koren, and R. P. Dwight. Non-intrusive uncertainty quantification using reduced cubature rules. *Journal of Computational Physics*, **332**:418–445, 2017. DOI: 10.1016/j.jcp.2016.12.011. arXiv: 1905.06177 [math.NA].

distribution with finite moments. It has positive weights and maximal polynomial degree. However, the nodes are not nested. The Gauss–Kronrod quadrature rule is an extension of a Gaussian quadrature rule, such that two nested rules with positive weights are obtained [103, 123, 185]. The Gauss–Kronrod–Patterson quadrature rule [124, 142] (or simply Gauss–Patterson quadrature rule) further extends this idea by repeatedly applying the same algorithm, such that a sequence of nested rules is obtained. However, it does not exist for any distribution [93, 94]. Even though many other extensions have been proposed over the years [67, 102, 115], in general it is difficult to obtain a series of nested quadrature rules with positive weights based on Gaussian rules [125]. Moreover often the smallest possible granularity between two consecutive nested quadrature rules can only be found by exhaustive search [22].

Another large group of well-known quadrature rules is formed by the Clenshaw–Curtis quadrature rules [35], or simply those quadrature rules that are based on Chebyshev approximations. The Clenshaw–Curtis rule is formed by the Chebyshev extrema and symbolic expressions of its nodes are known, i.e. see (2.16). Besides having excellent interpolation properties [84], it is well known that these quadrature rules have positive weights if the distribution under consideration is uniform (explicit expressions are known [180]). Moreover for non-uniform distributions, the condition number of the quadrature rule converges to unity [24]. However, the vanilla Clenshaw–Curtis nodes are only nested for exponentially growing numbers of nodes [79].

Both the Gaussian and Clenshaw–Curtis quadrature rules have explicitly predefined nodes based on the roots of orthogonal polynomials. This results in accurate quadrature rules, but the construction of an accurate nested quadrature rule with fine granularity based on these rules remains notoriously difficult.

The goal of this chapter is to propose a geometrical framework that mathematically describes the three elementary operations mentioned in the beginning of this chapter, i.e. the addition, removal, and replacement of nodes. All operations are based on the geometrical interpretation of the linear system describing the nodes and the weights [15, 41, 144], which yields necessary and sufficient conditions for a quadrature rule to have positive weights. The removal, addition, or replacement of a *single* node can be determined analytically, whereas numerical methods are required to determine all sequences of *multiple* nodes that can be added or replaced in a quadrature rule. The focus of this chapter is mainly on the mathematical aspects and not on the numerical construction of quadrature rules.

This chapter is structured as follows. In Section 3.2 the nomenclature and problem setting considered in this chapter is discussed, which is based on the notation introduced in Section 2.1.3. Then the three operations discussed above are considered. First, the removal of nodes is introduced in Section 3.3. It is always possible to remove a node from a quadrature rule such that the obtained rule has positive weights. By inverting the operation of removing a node, the addition of one or multiple nodes can be described. The problem of adding a single node can be solved analytically, which is done in Section 3.4. Contrary to the removal of nodes, it is not always possible to add a node to a quadrature rule such that the obtained rule has positive weights. Therefore the theory is extended to adding multiple nodes in Section 3.5, where the results developed for adding a single node will be used extensively. It is always possible to add multiple

nodes to a quadrature rule, provided that any number of nodes may be added to the rule. The replacement of one or multiple nodes follows from combining both operations, i.e. firstly adding a node and secondly removing a node. Any node in the quadrature rule can be replaced by a new node and all possible values of these new nodes can be described analytically. Section 3.6 contains numerical examples of quadrature rules to demonstrate simple applications of the proposed framework. The chapter is concluded in Section 3.7.

3.2. Preliminaries

The quadrature rule nomenclature relevant for this chapter is discussed in Section 3.2.1. The relevance of positive weights and the relation between positive weights and accuracy of a quadrature rule has been explained extensively in Section 2.1.3. However, in this chapter a slightly different form of the Lebesgue inequality from (2.12) is used, which is briefly considered in Section 3.2.2. The problem setting of this chapter is sketched mathematically in Section 3.2.3.

3.2.1. Nomenclature

Quadrature rules have been introduced extensively in Section 2.1.3 and only the concepts that are specifically necessary for this chapter are briefly repeated here. Since the focus is mainly on univariate quadrature rules, let $\Omega = [a,b] \subset \mathbb{R}$ with $-\infty \leq a < b \leq \infty$. In this chapter, the main interest is to approximate the weighted integral over a given continuous *univariate* function $u\colon \Omega \to \mathbb{R}$, i.e. to approximate the following operator:

$$\mathcal{I}u = \int_\Omega u(x)\,\rho(x)\,\mathrm{d}x = \int_a^b u(x)\,\rho(x)\,\mathrm{d}x.$$

Here, $\rho\colon \Omega \to [0,\infty)$ is a weighting function that is assumed to be known in this chapter. A quadrature rule approximates this integral by means of a weighted average, consisting of nodes and weights, denoted by $X_N = \{x_0,\dots,x_N\} \subset \Omega$ and $W_N = \{w_0,\dots,w_N\} \subset \mathbb{R}$ respectively*. The quadrature rule is the following operator $\mathcal{A}_N$:

$$\mathcal{A}_N u := \sum_{k=0}^{N} u(x_k)\,w_k \approx \mathcal{I}u.$$

As discussed previously, the consistency of this construction is measured by means of a polynomial space $\Phi_D = \operatorname{span}\{\varphi_0,\dots,\varphi_D\}$. In this chapter the focus is solely on univariate quadrature rules, so without loss of generality let $\varphi_k(x) = x^k$ for $k = 0,\dots,D$. Then due to linearity, the quadrature rule has degree D if

$$\mathcal{A}_N\varphi_k = \mathcal{I}\varphi_k, \text{ for all } k = 0,\dots,D. \tag{3.1}$$

This definition is only meaningful if ρ has finite moments, so that is assumed to be the case throughout this chapter.

*Since all quadrature rules constructed in this chapter are univariate, it holds that $X_N \subset \mathbb{R}$. Therefore the nodes are real numbers, denoted by x_k, and not multi-dimensional coordinate vectors, which are denoted by $\mathbf{x}_k$ as in the previous chapter.

The operators $\mathcal{A}_N$ and $\mathcal{I}$ are linear, so (3.1) defines a linear system that can be used to determine the weights, given the nodes and the moments of the distribution. In this case, the matrix of the linear system is the well-known Vandermonde matrix, denoted as follows:

$$\underbrace{\begin{pmatrix} x_0^0 & \cdots & x_N^0 \\ \vdots & \ddots & \vdots \\ x_0^N & \cdots & x_N^N \end{pmatrix}}_{V_N(X_N)} \begin{pmatrix} w_0 \\ \vdots \\ w_N \end{pmatrix} = \begin{pmatrix} \mu_0 \\ \vdots \\ \mu_N \end{pmatrix}, \tag{3.2}$$

with μ_j the raw moments of ρ:

$$\mu_j = \int_\Omega x^j \rho(x) \, \mathrm{d}x.$$

Throughout this chapter it is assumed that μ_j is known exactly for all j. The notation $V_D(X_N)$ denotes the Vandermonde matrix constructed using the quadrature rule nodes X_N and the polynomials up to degree D. Many quadrature rules constructed in this chapter have $D = N$, so we also use $V(X_N) := V_N(X_N)$.

It is well known that

$$\det V(X_N) = \prod_{0 \le i < j \le N} (x_j - x_i), \tag{3.3}$$

such that, given the nodes, (3.2) defines a unique solution of the weights provided that all nodes are distinct. A quadrature rule with distinct nodes that solves the linear system (3.2) (and therefore has degree N) is called interpolatory, as it can be formed by integrating the polynomial interpolant of u using the nodes X_N.

3.2.2. Accuracy of quadrature rules

In this chapter the focus is on constructing interpolatory quadrature rules with non-negative weights (which we will call with a little abuse of nomenclature a positive quadrature rule). An approximation of an integral by means of such a quadrature rule converges if the integrand is sufficiently smooth [148], which can among others be demonstrated by applying the Lebesgue inequality [24], provided that Ω is bounded. This inequality has been discussed in Chapter 2, see (2.12) on page 16. For the purposes of this chapter, the following form of the Lebesgue inequality is considered:

$$|\mathcal{A}_N u - \mathcal{I} u| \le (1 + \kappa_N) \, \mu_0 \inf_{\varphi \in \Phi_N} \|u - \varphi\|_\infty.$$

Here, κ_N is the *condition number* of the quadrature rule, see (2.13). In this chapter, only univariate interpolatory quadrature rules are constructed, so $D = N$ and moreover it is possible to explicitly determine $\inf_{\varphi \in \Phi_N} \|u - \varphi\|_\infty$ using the algorithm of Remez [188].

Arguably the best-known quadrature rule with positive weights is the Gaussian quadrature rule, a rule which is also considered in this chapter. Recall that the nodes of the Gaussian quadrature rule [74] are defined as the roots of the orthogonal polynomials with respect to the distribution ρ under consideration, e.g. Legendre polynomials for the uniform distribution, Jacobi polynomials for the Beta distribution, and Hermite

polynomials for the normal distribution. The uniquely defined rules always have positive weights and with $N+1$ nodes the rule has degree $2N+1$, however the rules are not nested.

The Gauss–Kronrod and Gauss–Patterson quadrature rules are extensions of Gaussian quadrature rules such that upon adding M nodes (with $M = N+2$ for the Gauss–Kronrod rule) to a rule of $N+1$ nodes, a (not necessarily positive) rule of degree $N+2M$ is obtained [103, 142]. The Patterson extension is also applicable to non-Gaussian quadrature rules, though possibly complex-valued nodes are obtained. The idea is to solve the following problem for $x_{N+1}, \ldots, x_{N+M}$, given quadrature rule nodes X_N:

$$\int_\Omega x^j \left(\prod_{k=0}^{N+M} (x - x_k) \right) \rho(x)\,\mathrm{d}x = 0, \text{ for } j = 0, \ldots, M-1. \tag{3.4}$$

Then the obtained rule has degree $N+2M$ [24, Theorem 5.1.3] and is defined uniquely. By construction, a Gaussian quadrature rule is obtained if $M = N+1$ (the weights of the nodes in X_N become zero). These rules are reobtained as a special case in the framework discussed in this chapter.

3.2.3. Problem setting

The problem studied in this chapter is how to remove nodes from, add nodes to, or replace nodes in a positive interpolatory quadrature rule such that it remains positive and interpolatory. Replacement of nodes is an immediate consequence of the addition of nodes. To keep the nomenclature concise, we will call these operations simply addition, removal, and replacement of nodes, without mentioning explicitly that a positive interpolatory quadrature rule should be obtained.

Removal

The removal of nodes can be formulated mathematically in the following way. If X_N and W_N are the nodes and weights of a positive interpolatory quadrature rule, the goal is to determine all nodes $x_{k_1}, \ldots, x_{k_M}$ such that $X_N \setminus \{x_{k_1}\}$ is a nodal set of a positive interpolatory quadrature rule. Hence this rule integrates all polynomials up to degree $N-1$ exactly. It will be demonstrated that there always exist exactly two nodes that can be removed from a quadrature rule, unless there are multiple nodes with zero weight.

The removal of multiple nodes is a straightforward extension: by repeatedly removing a single node from a quadrature rule, a sequence of nested quadrature rules is obtained. However, this construction cannot be used to determine *all* sequences of nodes that result in a positive interpolatory quadrature rule upon removal of all nodes of the sequence. Determining all such sequences requires incorporating negative weights and is studied in great detail in Chapter 4. In that chapter, the removal of nodes is further extended to accommodate the removal of multiple nodes, or more formally, all *sequences* of M nodes $X_M = \{x_{k_1}, \ldots, x_{k_M}\} \subset X_N$ (with M known) are determined such that $X_N \setminus X_M$ is a nodal set of a positive interpolatory quadrature rule.

Addition and replacement

To formulate addition mathematically, let a positive interpolatory quadrature rule X_N, W_N be given. Then the goal is to determine, for given M, all nodes such that the set X_{N+M} contains the nodes of a positive interpolatory quadrature rule and such that the

rules are nested, i.e. $X_N \subset X_{N+M}$. Moreover the number of nodes added to a quadrature rule should be minimal, so we are also interested in the minimal value of M such that a positive interpolatory quadrature rule with nodal set X_{N+M} exists.

If a positive quadrature rule is given that is not interpolatory, i.e. a quadrature rule such that $\mathcal{A}_N\varphi = \mathcal{I}\varphi$ for all $\varphi \in \mathbb{P}(K)$ with $K < N$, a positive interpolatory quadrature rule can be deduced from this rule by repeatedly removing nodes. Therefore without loss of generality addition and replacement are only considered for interpolatory quadrature rules.

The approach is to formulate, for given M, a necessary and sufficient condition for all M nodes that can be added. This condition can be used firstly to determine whether such nodes exist for a specific M and secondly to determine the nodes themselves. Moreover an immediate consequence is a mathematical formulation describing the replacement of nodes. It will be demonstrated that it is not always possible to add any number of nodes to an existing rule.

Adding M nodes with $M \geq N+1$ is always possible, as it is possible to add a Gaussian quadrature rule of $N+1$ nodes to a quadrature rule of $N+1$ nodes to obtain a positive interpolatory quadrature rule of $2N+2$ nodes. However, this yields a quadrature rule with $N+1$ weights equal to zero, so this construction is not of much practical interest.

The analysis is split into three sections. The removal of nodes is considered in Section 3.3, which yields an explicit description of all nodes that can be removed from a quadrature rule. By reversing the removal operation, the addition of a single node ($M = 1$) can be solved analytically and is discussed in Section 3.4. The addition of multiple nodes ($M > 1$) can only be done analytically for special cases. Based on the theory for $M = 1$, this problem is analyzed in Section 3.5.

3.3. Removal of nodes

In this section, a procedure is introduced to remove *one* node from a quadrature rule with positive weights. The procedure is such that the obtained quadrature rule (again) has positive weights and can be applied repeatedly to obtain a sequence of nested quadrature rules with positive weights, which we call a *reduced quadrature rule*. The idea derived in this section will be reversed in Section 3.4 to determine all nodes that can be added to a quadrature rule.

The procedure to remove nodes is inspired by the proof of Carathéodory's theorem. The obtained procedure that describes the removal of nodes is equivalent to previously obtained formulations [15, 38, 150, 194]. The main advantage of basing the procedure on Carathéodory's theorem is that existing theoretical results can be carried over to derive the exact number of nodes that can be removed. Moreover, the linear algebra framework can be used to accommodate the removal of multiple nodes, which is further considered in Chapter 4.

Firstly, Carathéodory's theorem is introduced in Section 3.3.1. Its constructive proof explicitly describes the nodes that can be removed. Secondly, some examples of reduced quadrature rules are presented in Section 3.3.2.

3.3.1. Carathéodory's theorem

To introduce the procedure, recall linear system (3.2):

$$\begin{pmatrix} x_0^0 & x_1^0 & \cdots & x_N^0 \\ x_0^1 & x_1^1 & \cdots & x_N^1 \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{N-1} & x_1^{N-1} & \cdots & x_N^{N-1} \\ x_0^N & x_1^N & \cdots & x_N^N \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-1} \\ w_N \end{pmatrix} = \begin{pmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{N-1} \\ \mu_N \end{pmatrix}, \tag{3.5}$$

which describes an interpolatory quadrature rule of degree N using the matrix $V_N(X_N)$. The goal is to find a subset of N nodes that again form an interpolatory quadrature rule. Such an interpolatory rule, with nodes X_{N-1}, is governed by the matrix $V_{N-1}(X_{N-1})$ and has degree $N-1$. The quadrature rule described by the system (3.2) can be interpreted as a rule of degree $N-1$, which is governed by the following system:

$$\begin{pmatrix} x_0^0 & x_1^0 & \cdots & x_N^0 \\ x_0^1 & x_1^1 & \cdots & x_N^1 \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{N-1} & x_1^{N-1} & \cdots & x_N^{N-1} \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-1} \\ w_N \end{pmatrix} = \begin{pmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{N-1} \end{pmatrix}. \tag{3.6}$$

This system is obtained by removing the last row of (3.5). The matrix of (3.6) is denoted as $V_{N-1}(X_N)$.

Each column of $V_{N-1}(X_N)$ is related to a node of the quadrature rule, so removing a column from the matrix above and solving the obtained system yields a nested quadrature rule of degree $N-1$. The question remains which column can be removed such that the system that remains has a solution with positive elements. The answer follows from (a variant of) the well-known theorem of Carathéodory. The constructive proof forms an algorithm to determine the columns that can be removed.

Theorem 3.1 (Carathéodory's theorem). *Let $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_N$ be $N+1$ vectors spanning an N-dimensional space. Let $\mathbf{v} = \sum_{k=0}^{N} \lambda_k \mathbf{v}_k$ with $\lambda_k \geq 0$. Then there exist $\beta_k \geq 0$ such that $\mathbf{v} = \sum_{k \in I} \beta_k \mathbf{v}_k$ and $I \subset \{0, \ldots, N\}$ with $|I| \leq N$.*

Proof. The vectors $\mathbf{v}_0, \ldots, \mathbf{v}_N$ are $N+1$ vectors in an N-dimensional space, so they must be linearly dependent. Therefore there exist c_k, not all equal to zero, such that

$$\sum_{k=0}^{N} c_k \mathbf{v}_k = 0.$$

Hence for any $\alpha \in \mathbb{R}$, it is true that

$$\begin{aligned} \mathbf{v} &= \sum_{k=0}^{N} \lambda_k \mathbf{v}_k - \alpha \sum_{k=0}^{N} c_k \mathbf{v}_k \\ &= \sum_{k=0}^{N} (\lambda_k - \alpha c_k) \mathbf{v}_k. \end{aligned}$$

Without loss of generality, we assume that at least one $c_k > 0$. Then the following choices are well defined:

$$k_0 = \underset{k=0,\dots,N}{\arg\min}\left(\frac{\lambda_k}{c_k} \mid c_k > 0\right) \text{ and } \alpha = \min_{k=0,\dots,N}\left(\frac{\lambda_k}{c_k} \mid c_k > 0\right) = \frac{\lambda_{k_0}}{c_{k_0}}. \tag{3.7}$$

If $\beta_k := \lambda_k - \alpha c_k$, it is true that $\beta_{k_0} = 0$ so with $I = \{0, 1, \dots, k_0 - 1, k_0 + 1, \dots, N\}$ the following holds:

$$\mathbf{v} = \sum_{k \in I} \beta_k \mathbf{v}_k. \qquad \blacksquare$$

The proof of Carathéodory's theorem can be exploited to remove columns from $V_{N-1}(X_N)$ or, equivalently, nodes from X_N preserving positive weights. To this end, let $\{\mathbf{v}_0, \dots, \mathbf{v}_N\}$ be the columns of $V_{N-1}(X_N)$ and rewrite (3.6) as follows:

$$\sum_{k=0}^{N} \mathbf{v}_k w_k = \boldsymbol{\mu}_{N-1}, \text{ for all } k = 0, \dots, N-1.$$

Here, $\boldsymbol{\mu}_{N-1} := (\mu_0, \dots, \mu_{N-1})^{\mathrm{T}}$. Following the proof of the theorem, there exist elements c_k (not all equal to zero) such that for all $\alpha \in \mathbb{R}$ it follows that

$$\sum_{k=0}^{N} \mathbf{v}_k (w_k - \alpha c_k) = \boldsymbol{\mu}_{N-1}, \text{ for all } k = 0, \dots, N-1.$$

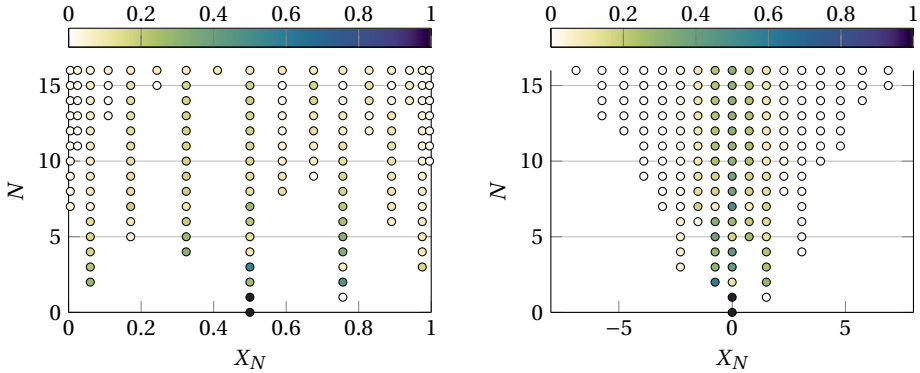
Notice that the elements c_k form a null vector $\mathbf{c} = (c_0, \dots, c_N)^{\mathrm{T}}$ of the matrix $V_{N-1}(X_N)$, which can be computed. Determining α and k_0 using (3.7) yields that $w_k - \alpha c_k \geq 0$ and that $w_{k_0} - \alpha c_{k_0} = 0$. Hence x_{k_0} can be removed from the quadrature rule, which yields an interpolatory quadature rule of degree $N-1$ with positive weights.

The removal step is not unique. The null vector $\mathbf{c}$ contains both positive and negative elements (guaranteed by the fact that the first row of the matrix contains only positive values), so $-\mathbf{c}$ is also a null vector with both positive and negative elements. Both of these null vectors can be used to eliminate a *different* node, provided that all weights are non-zero. In other words, given a null vector $\mathbf{c}$, there are two values of α that can be used in the proof: the value described by (3.7) or

$$\alpha = \max_{k=0,\dots,N}\left(\frac{\lambda_k}{c_k} \mid c_k < 0\right) = \frac{\lambda_{k_1}}{c_{k_1}}, \text{ with } k_1 = \underset{k=0,\dots,N}{\arg\max}\left(\frac{\lambda_k}{c_k} \mid c_k < 0\right). \tag{3.8}$$

This non-uniqueness will be exploited various times in subsequent chapters to construct quadrature rules with specific properties. Using the nomenclature of the problem setting as described in Section 3.2.3, the procedure yields two nodes $x_{k_0} \in X_N$ and $x_{k_1} \in X_N$ such that $X_N \setminus \{x_{k_0}\}$ and $X_N \setminus \{x_{k_1}\}$ are the nodes of positive interpolatory quadrature rules. If all weights are positive, it holds that $x_{k_0} \neq x_{k_1}$ (otherwise one of the values of α can be equal to 0).

It is demonstrated in Chapter 4 in a more general setting that these are the *only* two nodes that can be removed, i.e. the following theorem is proved.



(a) Reduced Gauss–Legendre **(b)** Reduced Gauss–Hermite

Figure 3.1: Reduced quadrature rules initiated with Gaussian quadrature rules consisting of 17 nodes. The Gauss–Hermite rule is constructed with respect to the standard normal distribution. The colors of the nodes indicate their weight.

3

Theorem 3.2 (Removal of nodes). *Let X_N and W_N form an interpolatory quadrature rule with $w_k > 0$ for all k. Consider the following nodal sets:*

$$X_{N-1}^{(k)} = X_N \setminus \{x_k\}, \text{ for } k = 0, \dots, N.$$

Then there exist k_0 and k_1 with $k_0 \neq k_1$ such that $X_{N-1}^{(k_0)}$ and $X_{N-1}^{(k_1)}$ form the nodes of positive interpolatory quadrature rules. Moreover, for $k \neq k_0$ and $k \neq k_1$, using $X_{N-1}^{(k)}$ as nodal set does not yield a positive quadrature rule.

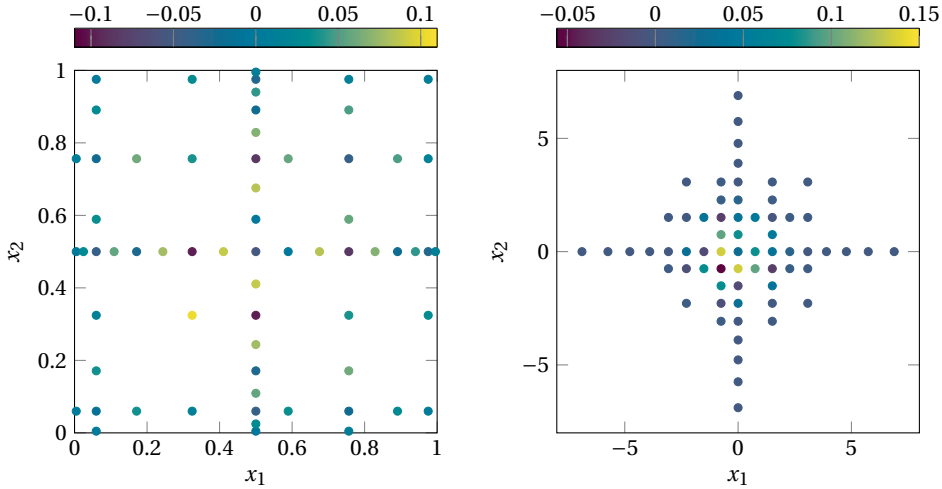
Proof. See the proof of Lemma 4.2 on page 81, which constitutes a slightly more general statement. ■

Notice that the removal step as discussed here can be applied straightforwardly to multivariate quadrature rules, since these rules can also be described by means of a single Vandermonde matrix. This is considered in Chapter 4. Moreover if the quadrature rule nodes and weights are given, no further knowledge about the distribution ρ, domain Ω, and moments μ_j is necessary (though obviously the moments can be reobtained from the quadrature rule).

3.3.2. Reduced Gaussian quadrature rules

The removal step can straightforwardly be applied to obtain a sequence of nested quadrature rules given a nodal set X_N. Doing so yields a sequence of nodal sets $X_N \supset X_{N-1} \supset \cdots \supset X_0$ such that each set describes an interpolatory quadrature rule with positive weights. As mentioned before, we call this the reduced quadrature rule.

Two examples of such sequences are depicted in Figure 3.1, where the initial rule is a Gaussian quadrature rule consisting of 17 nodes. In these examples the value of α, i.e. either (3.7) or (3.8), is selected such that the node that is closest to the center of



(a) Reduced Gauss–Legendre (65 nodes) **(b)** Reduced Gauss–Hermite (65 nodes)

Figure 3.2: Smolyak sparse grid constructed using quadrature rules determined by removal of nodes from a Gaussian quadrature rule. The colors of the nodes indicate their weight.

the domain (1/2 or 0) is preserved. All quadrature rules depicted in these figures have positive weights, are interpolatory, and together form a nested sequence of quadrature rules (though limited to at most 17 nodes).

As introduced in Chapter 2, the Smolyak sparse grid significantly benefits from using nested quadrature rules. The reduced quadrature rule is very suitable to construct Smolyak sparse grids with a small number of nodes, alleviating the rapid growth of the number of nodes of sparse grids generated with Gaussian quadrature rules. Moreover, a quadrature rule determined using a Smolyak sparse grid constructed with positive quadrature rules has a slowly growing condition number [135].

Smolyak sparse grids determined using reduced quadrature rules are depicted in Figure 3.2 (here the quadrature rules from Figure 3.1 are used), where those nested rules are used that coincide with the nested levels of the Clenshaw–Curtis quadrature rule. Compared to the Smolyak quadrature rules which were determined in Chapter 2 (see Figure 2.6 on page 21), the number of nodes is exactly the same as the sparse grid constructed using the Clenshaw–Curtis quadrature rule, though the reduced quadrature rules are not limited to the uniform distribution.

It is clearly visible that the reduced quadrature rules used to construct the sparse grids are asymmetric. The removal step can be extended to preserve symmetry, which can be used to construct sparse multivariate quadrature rules. However, this requires that the distributions of the input parameters are symmetric (such as the uniform and normal distribution used in Figure 3.2) and that the parameters are mutually independent. This is rarely the case for wind energy applications, so we will not consider such

rules in this thesis. Nonetheless, the interested reader is referred to Van den Bos et al. [15].

3.4. Addition of one node

Let X_N, W_N be a positive interpolatory quadrature rule. The goal is to determine all x_{N+1} such that $X_{N+1} = X_N \cup \{x_{N+1}\}$ forms the nodal set of a positive interpolatory quadrature rule, i.e. there exists a set of non-negative weights W_{N+1} such that

$$\sum_{k=0}^{N+1} x_k^j w_k^{(N+1)} = \mu_j, \text{ for } j = 0, \ldots, N+1.$$

Here, $w_k^{(N+1)}$ are the weights in the set W_{N+1} and μ_j is defined using (2.14) and assumed to be known for all j. Notice that in general W_N and W_{N+1} will completely differ, so we use the following notation for any N:

$$W_N = \{w_0^{(N)}, \ldots, w_N^{(N)}\}.$$

Moreover, with a little abuse of notation we will use $w_k^{(N)} = 0$ for all $k > N$.

In Section 3.4.1 we derive, based on the removal procedure outlined in the previous section, a necessary and sufficient condition for such an x_{N+1} to exist, which depends on the current nodes, weights, and moment μ_{N+1}. As such, the developed theory provides practical adjustments of a quadrature rule. These constitute addition and replacement of a node, without reducing the degree of the interpolatory quadrature rule. The details are discussed in Section 3.4.2 and will be very useful in the remainder of this thesis. In Section 3.4.3 the Patterson extension is discussed in light of the derived adjustments and some basic applications of the derived procedures are discussed, including the construction of a quadrature rule with positive weights.

3.4.1. Positive weight criterion

The key notion is that if the node x_{N+1} is given, it can be removed using the removal step outlined in Section 3.3. The removal step is based on a null vector $\mathbf{c} = (c_0, \ldots, c_{N+1})^{\mathrm{T}}$, which yields $w_k^{(N+1)} = w_k^{(N)} + c_k$ (for $k = 0, \ldots, N+1$). If this vector is such that $c_k \geq -w_k^{(N)}$, then $w_k^{(N+1)} \geq 0$, which is the primary goal. Deriving necessary and sufficient conditions on the vector $\mathbf{c}$ expressed in X_N and W_N is the focus of this section.

To this end, notice that the interpolatory quadrature rule X_N, W_N has degree N, so after adding x_{N+1} the following should hold to ensure that the new rule is interpolatory:

$$\mu_j = \sum_{k=0}^{N} x_k^j w_k^{(N)} = \sum_{k=0}^{N+1} x_k^j w_k^{(N+1)}, \text{ for } j = 0, \ldots, N.$$

From this, it follows for $j = 0, \ldots, N$ that (using $w_{N+1}^{(N)} = 0$):

$$0 = \sum_{k=0}^{N+1} x_k^j w_k^{(N+1)} - \sum_{k=0}^{N+1} x_k^j w_k^{(N)} = \left(\sum_{k=0}^{N+1} x_k^j w_k^{(N)} + \sum_{k=0}^{N+1} x_k^j c_k \right) - \sum_{k=0}^{N+1} x_k^j w_k^{(N)} = \sum_{k=0}^{N+1} x_k^j c_k. \quad (3.9)$$

This equation describes that $\mathbf{c}$ is a null vector of the matrix $V_N(X_{N+1})$. The goal is to construct X_{N+1} and W_{N+1} such that they form a quadrature rule of degree $N+1$. Hence with $\mu_{N+1} = \int_\Omega x^{N+1}\,\rho(x)\,\mathrm{d}x$ given, it should hold that

$$\sum_{k=0}^{N+1} x_k^{N+1} w_k^{(N+1)} = \mu_{N+1},$$

which can be expressed in terms of the vector $\mathbf{c}$ as

$$\varepsilon_{N+1} := \mu_{N+1} - \sum_{k=0}^{N} x_k^{N+1} w_k^{(N)} = \sum_{k=0}^{N+1} x_k^{N+1} c_k. \tag{3.10}$$

The value of ε_{N+1} can be interpreted as the approximation error of the quadrature rule with nodes X_N and weights W_N with respect to μ_{N+1}. Combining (3.9) and (3.10) yields the following system of linear equations for the vector $\mathbf{c}$:

$$\begin{pmatrix} x_0^0 & \cdots & x_N^0 & x_{N+1}^0 \\ \vdots & \ddots & \vdots & \vdots \\ x_0^N & \cdots & x_N^N & x_{N+1}^N \\ x_0^{N+1} & \cdots & x_N^{N+1} & x_{N+1}^{N+1} \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_N \\ c_{N+1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \varepsilon_{N+1} \end{pmatrix},$$

or more compactly:

$$V(X_{N+1})\mathbf{c} = \boldsymbol{\varepsilon},$$

with $\boldsymbol{\varepsilon} = (0,\ldots,0,\varepsilon_{N+1})^{\mathrm{T}}$. The vector $\boldsymbol{\varepsilon}$ has a large number of zeros, so it is convenient to apply Cramer's rule to this linear system, which yields

$$c_k = \frac{\det V_{(k)}(X_{N+1})}{\det V(X_{N+1})},$$

where $V_{(k)}(X_{N+1})$ is equal to $V(X_{N+1})$ with the k-th column replaced by $\boldsymbol{\varepsilon}$, where the indexing of columns is started with 0. This expression can be simplified by noticing that

$$\begin{aligned} \det V_{(k)}(X_{N+1}) &= (-1)^{(N+2)+(k+1)} \varepsilon_{N+1} \det V(X_{N+1} \setminus \{x_k\}) \\ &= (-1)^{N+k+1} \varepsilon_{N+1} \det V(X_{N+1} \setminus \{x_k\}), \end{aligned}$$

with $V(X_{N+1} \setminus \{x_k\})$ the $(N+1)\times(N+1)$ Vandermonde matrix constructed with the nodal set $X_{N+1} \setminus \{x_k\}$. By using the Vandermonde determinant from (3.3), the following is

obtained for $k = 0, \ldots, N+1$:

$$
\begin{aligned}
c_k = \frac{\det V_{(k)}(X_{N+1})}{\det V(X_{N+1})} &= (-1)^{N+k+1}\varepsilon_{N+1}\frac{\det V(X_{N+1}\setminus\{x_k\})}{\det V(X_{N+1})} \\
&= (-1)^{N+k+1}\varepsilon_{N+1}\left(\prod_{\substack{0\le i<j\le N+1\\ i,j\neq k}}(x_j - x_i)\right)\Bigg/\left(\prod_{0\le i<j\le N+1}(x_j - x_i)\right) \\
&= (-1)^{N+k+1}\varepsilon_{N+1}\Bigg/\left(\prod_{0\le i<k}(x_k - x_i)\prod_{k<j\le N+1}(x_j - x_k)\right) \\
&= (-1)^{N+1}\varepsilon_{N+1}\Bigg/\left(\prod_{\substack{j=0\\ j\neq k}}^{N+1}(x_j - x_k)\right) \\
&= \varepsilon_{N+1}\Bigg/\left(\prod_{\substack{j=0\\ j\neq k}}^{N+1}(x_k - x_j)\right). \qquad (3.11)
\end{aligned}
$$

The denominator of this expression can be written as $L'_N(x_k)$, where $L_N(x) = \prod_{j=0}^{N}(x - x_j)$ is the nodal polynomial. To keep the dependence on x_{N+1} clear, this notation is used only sparingly in this chapter.

The goal is to have positive weights, i.e. $w_k^{(N+1)} = w_k^{(N)} + c_k \ge 0$, which proves the following theorem.

Theorem 3.3 (Addition of one node). *Let X_N, W_N form an interpolatory quadrature rule. Then $X_{N+1} = X_N \cup \{x_{N+1}\}$ forms the nodal set of a positive interpolatory quadrature rule if and only if*

$$
-\varepsilon_{N+1}\Bigg/\left(\prod_{\substack{j=0\\ j\neq k}}^{N+1}(x_k - x_j)\right) \le w_k^{(N)}, \text{ for } k = 0, \ldots, N+1. \qquad (3.12)
$$

If $\varepsilon_{N+1} = 0$, i.e. $\mathcal{A}_N x^{N+1} = \mu_{N+1}$, then the theorem yields that the new rule has positive weights if and only if the current rule has positive weights. This is not surprising: any node x_{N+1} can be added to such a rule with $w_{N+1}^{(N+1)} = 0$ (and with $w_k^{(N+1)} = w_k^{(N)}$ for $k = 0, \ldots, N$).

From a computational point of view (3.12) might not be a numerically stable way of computing the bounds that describe all nodes that can be added. In the context of quadrature rules, numerical instabilities are usually alleviated by changing the basis of the Vandermonde matrix, but this is not applicable in this case since the determinant is up to a sign independent of the basis used to construct the Vandermonde matrix. Nonetheless, (3.12) can be evaluated in a numerical stable way using the well-known barycentric formulation of the interpolating polynomial. It is briefly considered in Chapter 6 (see (6.4) on page 127), but a more elaborate explanation can be found in Berrut and Trefethen [7].

3.4.2. Quadrature rule adjustments

Theorem 3.3 describes a necessary and sufficient condition for a quadrature rule to have positive weights if both x_{N+1} and ε_{N+1} are known. A main novelty of this chapter is to employ a geometrical interpretation of (3.12), from which several possible adjustments of quadrature rules can be derived. The most straightforward one is that all nodes x_{N+1} can be determined that yield a positive interpolatory quadrature rule upon adding one of them to an existing quadrature rule. Moreover the formula also yields procedures to replace nodes in a quadrature rule, keeping the weights positive. The latter adjustment will be useful in Section 3.5, where it can be used to determine all possible M nodes that can be added to a rule.

We proceed by further simplifying (3.12) and discussing the geometrical relation between the new node x_{N+1} and the quadrature error ε_{N+1}. This geometrical relation yields a constructive description of the addition and replacement of nodes in a positive interpolatory quadrature rule such that positivity of the weights is preserved.

Geometry of nodal addition

The inequalities from (3.12) are $N+2$ linear inequalities in x_{N+1} and ε_{N+1}. This can be seen by rewriting (3.11) as follows:

$$c_k \prod_{\substack{j=0\\ j\neq k}}^{N+1} (x_k - x_j) = \varepsilon_{N+1}, \text{ for } k = 0,\dots,N+1. \tag{3.13}$$

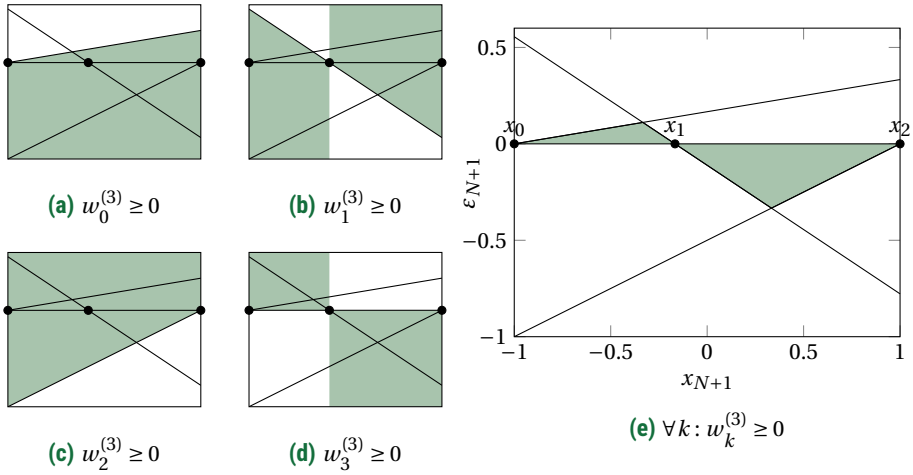
If two values of x_{N+1}, c_k (for $k = 0,\dots,N+1$), or ε_{N+1} are known, all other values can be determined from these expressions, which enforces that the obtained quadrature rule is again interpolatory. To incorporate positive weights, we use that for $k = 0,\dots,N$ it holds that

$$\varepsilon_{N+1} = c_k \prod_{\substack{j=0\\ j\neq k}}^{N+1} (x_k - x_j) = (x_k - x_{N+1})\, c_k \underbrace{\prod_{\substack{j=0\\ j\neq k}}^{N} (x_k - x_j)}_{\text{Independent of } x_{N+1}}.$$

By combining this with (3.12) and requiring $w_k^{(N)} + c_k \geq 0$ inequalities of the following form are obtained:

$$\begin{aligned} \varepsilon_{N+1} &\leq -w_k^{(N)} (x_k - x_{N+1}) \prod_{\substack{j=0\\ j\neq k}}^{N} (x_k - x_j) \quad \text{if } \prod_{\substack{j=0\\ j\neq k}}^{N+1} (x_k - x_j) \leq 0, \\ \varepsilon_{N+1} &\geq -w_k^{(N)} (x_k - x_{N+1}) \prod_{\substack{j=0\\ j\neq k}}^{N} (x_k - x_j) \quad \text{if } \prod_{\substack{j=0\\ j\neq k}}^{N+1} (x_k - x_j) \geq 0. \end{aligned} \tag{3.14}$$

These are linear inequalities describing the relation between x_{N+1} and ε_{N+1} such that $w_k^{(N+1)} \geq 0$ for $k = 0,\dots,N$. For $k = N+1$ it holds that $w_k^{(N)} = 0$, so by using that $c_k =$



(a) $w_0^{(3)} \geq 0$ **(b)** $w_1^{(3)} \geq 0$ **(c)** $w_2^{(3)} \geq 0$ **(d)** $w_3^{(3)} \geq 0$ **(e)** $\forall k : w_k^{(3)} \geq 0$

Figure 3.3: The quadrature rule error ε_{N+1} versus the new node x_{N+1} using the quadrature rule $X_N = \{-1, -1/6, 1\}$ and $\rho \equiv 1/2$. The solid lines depict locations such that one weight becomes zero. *Left:* Regions where individual weights are positive. *Right:* Region where all weights are positive, which is the intersection of the left figures.

$w_k^{(N+1)}$, (3.13) translates to:

$$\begin{aligned} \varepsilon_{N+1} \leq 0 \quad &\text{if } \prod_{j=0}^{N} (x_{N+1} - x_j) \leq 0, \\ \varepsilon_{N+1} \geq 0 \quad &\text{if } \prod_{j=0}^{N} (x_{N+1} - x_j) \geq 0. \end{aligned} \tag{3.15}$$

Even though the rightmost inequalities are non-linear, their sign solely depends on the location of x_{N+1} with respect to the other nodes. Hence the exact value of the product is not of importance.

These inequalities are visualized as functions from x_{N+1} to ε_{N+1} in Figure 3.3 for the quadrature rule with X_N and W_N as follows:

$$X_N = \left\{-1, -\frac{1}{6}, 1\right\}, W_N = \left\{\frac{1}{10}, \frac{24}{35}, \frac{3}{14}\right\}.$$

This is an (obviously positive) interpolatory quadrature rule with $\Omega = [-1, 1]$ and $\rho \equiv 1/2$. The solid lines in the figures depict all $(x_{N+1}, \varepsilon_{N+1})$ pairs such that one weight becomes equal to zero (i.e. where equality is attained in inequality (3.14) or (3.15)). The region where individual weights are positive are shaded in subfigures 3.3a, 3.3b, 3.3c, and 3.3d. Subfigure 3.3e is the intersection of these figures and therefore depicts regions where all weights are positive. Any $(x_{N+1}, \varepsilon_{N+1})$ pair in the shaded region describes a positive interpolatory quadrature rule that contains the original three nodes.

The left subfigures demonstrate some key properties of the derived inequalities. The inequalities are linear and switch sign at the node, which is the rightmost condition of (3.14). The characteristics of the last inequality (subfigure 3.3d) solely depend on the location of x_{N+1} with respect to the other nodes. A combination of all inequalities (subfigure 3.3e) has varying characteristics between different nodes, but it is always a system of linear inequalities. The line $\varepsilon_{N+1} = 0$ is contained in all shaded regions, because any node with weight equal to zero can be added to the rule if the next moment μ_{N+1} is already correctly integrated by the quadrature rule.

These figures can be interpreted in two ways. Firstly, if a new node x_{N+1} is given, an upper bound and lower bound on ε_{N+1} can be determined such that upon adding x_{N+1} to the quadrature rule, a positive interpolatory quadrature rule is obtained. Geometrically these are the bounds of the shaded area with the $x = x_{N+1}$ line. This interval is never empty (as $\varepsilon_{N+1} = 0$ is always in the shaded region). Secondly, if ε_{N+1} is given, a (possibly empty) set can be determined such that a positive interpolatory quadrature rule is obtained upon adding a node from such a set. Geometrically this is equivalent to determining the bounds of the shaded area with the $y = \varepsilon_{N+1}$ line.

The second interpretation can be used to add nodes to a quadrature rule, i.e. ε_{N+1} is known and the goal is to determine x_{N+1}. The first interpretation can be used to replace nodes within a quadrature rule: x_{N+1} is added to the node and an existing node can be removed by setting its weight to zero.

Addition of a node

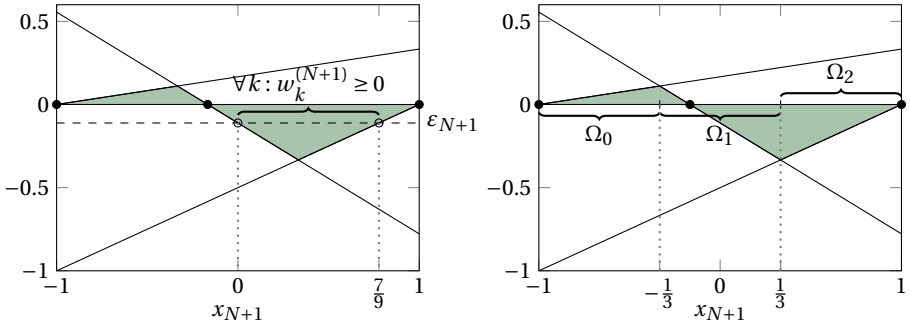
Adding a node is determining an x_{N+1} that solves (3.14) and (3.15) if ε_{N+1} is known. This is equivalent to combining the solutions $x_{N+1}^{[k]}$ (indexed by $[k]$ with $k = 0, \ldots, N$) of the following problems:

$$\varepsilon_{N+1} = -w_k^{(N)}(x_k - x_{N+1}^{[k]}) \underbrace{\prod_{\substack{j=0 \\ j\neq k}}^{N} (x_k - x_j)}_{L'_N(x_k)}, \text{ for } k = 0, \ldots, N, \tag{3.16}$$

hence if $w_k^{(N)} \neq 0$:

$$x_{N+1}^{[k]} = \frac{\varepsilon_{N+1} + w_k^{(N)} x_k L'_N(x_k)}{w_k^{(N)} L'_N(x_k)}, \text{ for } k = 0, \ldots, N.$$

Here we used L'_N to make the notation more compact. The nodes $x_{N+1}^{[k]}$ are such that, if added to the quadrature rule, a (possibly negative) interpolatory quadrature rule is obtained with $w_k^{(N+1)} = 0$. Those $x_{N+1}^{[k]}$ that yield positive quadrature rules describe intervals and all nodes in these intervals can be added to the rule such that positive weights are obtained. The bounds of these intervals, i.e. the solutions of (3.16), are depicted in Figure 3.4a as open circles. The same quadrature rule example as in the previous section is used and a constant valued ρ is considered. In this case, the three solutions are (from left to right) $-5/3$, 0, and $7/9$, of which the first is not visible in the figure. Adding any of these nodes yields a quadrature rule with positive weights,



(a) Addition ($\varepsilon_{N+1} = -1/9$) **(b)** Replacement

Figure 3.4: Addition of a new node to and replacement of an existing node within the quadrature rule $X_N = \{-1, -1/6, 1\}$ and $\rho \equiv 1/2$.

but we emphasize that this is generally not the case for other quadrature rules. Hence adding any node from the intervals $(-\infty, -5/3]$ or $[0, 7/9]$ yields a positive interpolatory quadrature rule. Restricting x_{N+1} to the set Ω further reduces the number of possible intervals.

The node $x_{N+1}^{[k]}$ does not exist if $\varepsilon_{N+1} \neq 0$ and $w_k^{(N)} = 0$. This can be derived mathematically, but it also follows from the mere fact that all weights change (see (3.16)) upon addition of a node to a quadrature rule, so $w_k^{(N+1)} = w_k^{(N)} = 0$ is not possible. If $\varepsilon_{N+1} = 0$, no node can be added to enforce that $w_k^{(N)} = 0$. However, any node with weight equal to zero can be added, hence the formula yields $x_{N+1}^{[k]} = x_k$ with $w_{N+1}^{(N+1)} = 0$. Technically, the quadrature rule now has a node equal to x_k with weight equal to zero. Nonetheless, this results in a singular Vandermonde matrix (which contradicts the theory developed so far), so we do not further study this specific case.

If $\Omega = \mathbb{R}$ and the number of nodes is odd, it is always possible to add a single node to a quadrature rule. Geometrically this means that the leftmost and rightmost shaded regions grow to infinity and minus infinity respectively (or vice versa). Similarly, if $\Omega = \mathbb{R}$ and the number of nodes is even, it is always possible to add a single node if $\varepsilon_{N+1} \geq 0$. However, in any other case (i.e. that of a bounded Ω or even number of nodes with $\varepsilon_{N+1} < 0$) adding a single node to a quadrature rule is not always possible. An example is the following interpolatory quadrature rule:

$$X_N = \left\{-1, -\frac{1}{6}, \frac{1}{11}, 1\right\}, W_N = \left\{\frac{29}{180}, \frac{144}{595}, \frac{1331}{3060}, \frac{17}{105}\right\}.$$

Note that this example can be obtained straightforwardly by adding the node $1/11$ to the previous stated quadrature rule and redetermining the weights likewise.

Replacement of a node

The replacement of a single node can be interpreted as the addition of a node and subsequently the removal of a (different) node or as adding a node such that an existing weight becomes zero.

The first interpretation follows by adding a node to the quadrature rule such that the added node has non-zero weight. Geometrically, this is selecting a $(x_{N+1}, \varepsilon_{N+1})$ in the shaded region with $\varepsilon_{N+1} \neq 0$. Then by Theorem 3.2 there exist two nodes that can be removed, with at least one node (either x_{k_0} or x_{k_1}) not equal to x_{N+1}. Or equivalently, both $X_N \setminus \{x_{k_0}\} \cup \{x_{N+1}\}$ and $X_N \setminus \{x_{k_1}\} \cup \{x_{N+1}\}$ yield positive interpolatory quadrature rules and one of these quadrature rules is not equal to the quadrature rule with nodes X_N.

The second interpretation follows by noticing that replacing a node is equivalent to adding a node, with the difference that the goal is to determine this node such that the weight of an existing node in the obtained quadrature rule becomes zero, i.e. $w_k^{(N+1)} = 0$ for a $k \leq N$. This is equivalent to determining a specific $(x_{N+1}, \varepsilon_{N+1})$ pair that yields $w_k^{(N+1)} = 0$, which was used previously to determine all possible additions. The main difference with addition is that the next moment μ_{N+1} is not used, as the number of nodes and the degree of the rule do not change. This allows to freely "choose" ε_{N+1}.

Using the latter interpretation, an explicit relation can be deduced that describes the replacement of any node $x_k \in X_N$. It follows from the derived relation between ε_{N+1} and x_{N+1}. So by reconsidering (3.16) with the goal to determine both x_{N+1} and all $\varepsilon_{N+1}^{[k]}$ (indexed by $[k]$ with $k = 0, \ldots, N$) that make $w_k^{(N+1)} = 0$ the following expressions are obtained:

$$\varepsilon_{N+1}^{[k]} = -w_k^{(N)}(x_k - x_{N+1}) \prod_{\substack{j=0 \\ j \neq k}}^{N} (x_k - x_j), \text{ for } k = 0, \ldots, N. \tag{3.17}$$

We will interpret this expression as a function of x_{N+1}, denoted by $\varepsilon_{N+1}^{[k]} : \Omega \to \mathbb{R}$. These functions represent the solid lines in Figure 3.4b. By using $\varepsilon_{N+1} = \varepsilon_{N+1}^{[k]}(x_{N+1})$, a positive interpolatory quadrature rule with $w_k^{(N+1)} = 0$ is obtained upon adding x_{N+1} to the rule.

It follows that for every $x_{N+1} \in \Omega$ there is an $x_k \in X_N$ such that the quadrature rule with nodes $X_N \cup \{x_{N+1}\} \setminus \{x_k\}$ is positive and interpolatory. The node x_k can be found by determining the bounds on the shaded region and observing which node belongs to the obtained bound. The sets Ω_k describe this property mathematically: if an $x_{N+1} \in \Omega_k$ is added to the quadrature rule, then $X_N \cup \{x_{N+1}\} \setminus \{x_k\}$ forms the nodal set of a positive interpolatory quadrature rule. The fact that $\Omega = \bigcup_{k=0}^{N} \Omega_k$ follows from Figure 3.4b. Mathematically, these sets can be denoted (possibly less intuitively) in the following way:

$$x_{N+1} \in \Omega_k \iff \begin{cases} \varepsilon_{N+1}^{[k]}(x_{N+1}) = \min_j \left(\varepsilon_{N+1}^{[j]}(x_{N+1}) \mid \varepsilon_{N+1}^{[j]}(x_{N+1}) \geq 0 \right) & \text{if } \prod_{j \neq k}^{N+1} (x_k - x_j) \leq 0, \\ \varepsilon_{N+1}^{[k]}(x_{N+1}) = \max_j \left(\varepsilon_{N+1}^{[j]}(x_{N+1}) \mid \varepsilon_{N+1}^{[j]}(x_{N+1}) \leq 0 \right) & \text{if } \prod_{j \neq k}^{N+1} (x_k - x_j) \geq 0. \end{cases} \tag{3.18}$$

The intersection of the lines in Figure 3.4b forms a special case. Notice that Ω_k is a closed interval, because $\prod_{j \neq k}^{N} (x_k - x_j) = 0$ forms the boundary and is included in the interval. Combining this with $\Omega = \bigcup_{k=0}^{N} \Omega_k$ yields that for each Ω_k there exists at least one other Ω_l such that $\Omega_k \cap \Omega_l$ is not empty. This means that for $x_{N+1} \in \Omega_k \cap \Omega_l$, the quadrature rule $X_N \cup \{x_{N+1}\} \setminus \{x_k\} \setminus \{x_l\}$ is positive, interpolatory, and has degree N, even though it

consists only of N nodes. The latter result is remarkable: two nodes are removed and one is added, but the degree of the quadrature rule is not affected. An example of such a rule would be $X_N = \{-1, 1/3\}$, obtained by adding $1/3$ to the quadrature rule example of Figure 3.4b. Such rules have a non-trivial high degree and are therefore more accurate than interpolatory quadrature rules without this property.

This special case can be further generalized by allowing negative weights, which will be very useful in light of multiple node addition discussed in Section 3.5. To introduce this formally, let the node $x_{(k,l)}$ be such that $X_N \cup \{x_{(k,l)}\} \setminus \{x_k\} \setminus \{x_l\}$ is a (possibly negative) interpolatory quadrature rule. This node exists if and only if it satisfies the following linear equality, that follows from (3.17):

$$-w_k^{(N)}(x_k - x_{(k,l)}) \prod_{\substack{j=0 \\ j\neq k}}^{N} (x_k - x_j) = -w_l^{(N)}(x_l - x_{(k,l)}) \prod_{\substack{j=0 \\ j\neq l}}^{N} (x_l - x_j). \tag{3.19}$$

Notice that this linear inequality does not necessarily have a solution. Graphically there is no solution if the lines from Figure 3.4b through x_k and x_l are parallel.

The node $x_{(k,l)}$ is independent of x_k and x_l. This is not directly evident, as (3.19) depends on these nodes. However, it can be demonstrated by using that the rule is interpolatory, which yields:

$$w_k^{(N)} = \int_\Omega \ell_k(x)\,\rho(x)\,\mathrm{d}x = \frac{1}{L_N'(x_k)} \int_\Omega \frac{L_N(x)}{x - x_k}\rho(x)\,\mathrm{d}x, \text{ with } \ell_k(x) = \prod_{\substack{j=0 \\ j\neq k}}^{N} \frac{x - x_j}{x_k - x_j}.$$

This expression follows straightforwardly from applying (2.10) (see page 15). Here, $\ell_k(x)$ is the univariate k-th Lagrange basis polynomial from (2.5) (see page 13). Replacing this expression in (3.19) and using that $L_N'(x_k) = \prod_{j\neq k}(x_k - x_j)$ yields an equality that can be simplified to the following:

$$x_{(k,l)} = \left(\int_\Omega x\,L_{(k,l)}(x)\,\rho(x)\,\mathrm{d}x\right) \Big/ \left(\int_\Omega L_{(k,l)}(x)\,\rho(x)\,\mathrm{d}x\right), \text{ with } L_{(k,l)}(x) = \prod_{\substack{j=0 \\ j\neq k,l}}^{N} (x - x_j).$$

This expression is in fact a Patterson extension (consider (3.4) with $j = 0$). The tight relation between the Patterson extension and the framework discussed in this chapter is further discussed later in this chapter.

3.4.3. Constructing quadrature rules

In the previous section the theoretical foundation for extending a positive interpolatory quadrature rule with a single node is derived. In this section, firstly it is discussed how addition relates naturally to the Patterson extension [141, 142] of (non-Gaussian) quadrature rules. Secondly, due to the simplicity of addition and replacement of a node, quadrature rules based on these procedures can be derived numerically fast and accurately. Two examples of such rules are discussed.

As discussed previously, there does not always exist a single node that can be added such that positive weights are obtained, so it is not trivial to construct a sequence of

positive interpolatory quadrature rules. There are various possibilities to alleviate this, e.g. by allowing negative weights, relaxing the strict requirement that all nodes have to be preserved, or by adding multiple nodes instead of one.

Firstly, if it is not possible to add a node to the quadrature rule such that positive weights are preserved, a heuristic can be used to select the node instead. As example, Leja nodes [106] are considered, which form a sequence of nodes constructed originally for the purpose of polynomial interpolation. Leja nodes will be used extensively in Chapter 6 and embed straightforwardly in the derived framework of nodal addition. Secondly, a quadrature rule based on the *replacement* of nodes is presented. This rule has positive weights and is interpolatory, but is strictly speaking not "fully" nested. The addition of multiple nodes is further considered in Section 3.5.

Patterson extension

Remarkably, both the addition and replacement of a node can yield a Patterson extension of a quadrature rule. In both cases, the focus is on the nodes that yield a zero weight upon addition to the quadrature rule.

In Section 3.4.2 it was noticed that any weight from a quadrature rule can be made equal to zero by exploiting the relation between ε_{N+1} and x_{N+1}. As example, the quadrature rule $X_N = \{-1, -1/6, 1\}$ was considered, where the nodes $-5/3$, 0, and $7/9$ are such that upon adding one of these to the rule, a rule of only three nodes with non-zero weights of degree three is obtained. Notice that these nodes are Patterson extensions of quadrature rules (as discussed in Section 3.2.2), as they can be interpreted as adding one node ($M = 1$) to a quadrature rule of two nodes ($N = 1$), obtaining a rule of degree three ($N + 2M = 3$). This also holds in general: for given k, adding one node $x_{N+1}^{[k]}$ from (3.16) (so $M = 1$) to the interpolatory quadrature rule $X_N \setminus \{x_k\}$ (with degree $N-1$) yields a quadrature rule with $N+1$ nodes and degree $N+1$ (which equals $(N-1)+2M$).

Recall that the notation $x_{(k,l)}$ was introduced to denote nodes that, upon adding them to the rule, yield a (possibly negative) interpolatory quadrature rule with $w_k^{(N+1)} = w_l^{(N+1)} = 0$. These nodes also form a Patterson extension. To see this, notice that the replacement is adding a single node to the quadrature rule $X_{N-2} = X_N \setminus \{x_l\} \setminus \{x_k\}$. The Patterson extension of a single node of this quadrature rule is a quadrature rule consisting of N nodes of degree $(N-2)+2M = N$ (adding one node means $M = 1$). By construction, this rule has the nodes $X_{N-2} \cup \{x_{(k,l)}\}$.

Reconsider for example the quadrature rule with the nodes $X_N = \{-1, -1/6, 1\}$ and $\rho \equiv 1/2$. Then it is straightforward to determine using (3.19) that $x_{(0,1)} = -1/3$, $x_{(0,2)} = 2$, and $x_{(1,2)} = 1/3$. Hence these are three Patterson extensions of the quadrature rule nodes $\{1\}$, $\{-1/6\}$, and $\{-1\}$. Indeed, the quadrature rules with the nodes $\{-1/3, 1\}$, $\{-1/6, 2\}$, or $\{-1, 1/3\}$ have degree equal to 2.

Notice that $x_{(k,l)}$ is not a Patterson extension of the quadrature rule that has been used to determine it, i.e. X_N, W_N in (3.19). However, its definition allows for a straightforward way to determine this extension. First, add (randomly) two nodes to the quadrature rule X_N, W_N, obtaining a possibly negative interpolatory quadrature rule X_{N+2}, W_{N+2}. Then the node $x_{(N+1,N+2)}$ is the Patterson extension of the quadrature rule with nodes X_N, because upon adding this node to X_{N+2}, the weights of the randomly added nodes become zero. As the Patterson extension is unique, this construction is well defined.

Naturally, this is not the preferred approach to construct a Patterson extension, but it embeds such extensions into the framework discussed here.

The Patterson extension is also obtained as a special case if multiple nodes are added to a quadrature rule. This will be discussed in Section 3.5.3.

Leja nodes

Leja nodes [106] are nodes obtained by heuristic optimization. Their inductive definition is as follows. Let $x_0, \ldots, x_N$ be a sequence of Leja nodes with $x_k \in \Omega$ for all k. Then x_{N+1} is obtained as follows:

$$x_{N+1} = \operatorname*{arg\,max}_{x \in \Omega} \prod_{k=0}^{N} |x - x_k|.$$

This definition is only meaningful if Ω is bounded. Leja nodes can be extended to incorporate ρ and unbounded Ω, obtaining *weighted Leja nodes*, which will be considered in great detail in Chapter 6.

The sequence of nodes obtained by this definition has been used extensively for the purpose of interpolation [130, 133]. It is common to initiate the sequence with $x_0 = a$ or $x_0 = b$ (with $\Omega = [a, b]$). It has been demonstrated that the Lebesgue constant of Leja nodes, as defined by (2.9), grows sub-exponentially [178], which implies that a polynomial interpolant constructed with this sequence converges for any analytic function (for $N \to \infty$). However, Leja nodes do generally not form an interpolatory quadrature rule with positive weights.

The insight gained from the framework of this chapter is that Leja nodes can be interpreted as greedily minimizing the absolute value of the weight of the new node, which is an approximate measure of the condition number of the quadrature rule (since all weights add to unity). Since this number is a measure for the accuracy of the quadrature rule and ensures numerical stability (recall its definition on page 17), ideally quadrature rules have small condition number. To see this, recall the definition of c_{N+1} from (3.13):

$$c_{N+1} \prod_{k=0}^{N} (x_{N+1} - x_k) = \varepsilon_{N+1}.$$

Using that $w_{N+1}^{(N+1)} = c_{N+1}$, it follows that

$$|w_{N+1}^{(N+1)}| = |\varepsilon_{N+1}| \Bigg/ \left(\prod_{k=0}^{N} |x_{N+1} - x_k| \right).$$

By definition, the Leja node x_{N+1} is such that it maximizes the denominator of this expression. Hence Leja nodes are such that $|w_{N+1}^{(N+1)}|$ is minimal, although this does not guarantee positive weights.

Partially nested, positive, and interpolatory quadrature rule

The addition and replacement of a single node are straightforward procedures described as the solutions of linear inequalities. However, there does not always exist a single node that can be added such that all weights remain positive. In this section, this is alleviated by relaxing the requirement that $X_N \subset X_{N+M}$.

To this end, let X_N and $\widehat{X}_{N+1}$ be the nodes of two positive interpolatory quadrature rules, possibly with $X_N \not\subset \widehat{X}_{N+1}$. The nodes $\widehat{X}_{N+1}$ can for example form a Gaussian quadrature rule. The idea is to iteratively replace nodes in $\widehat{X}_{N+1}$ with nodes from X_N, i.e. removing $x_k \in \widehat{X}_{N+1}$ and adding $x_k \in X_N$. Ideally, all nodes $x_k \in X_N$ can be added to $x_k \in \widehat{X}_{N+1}$, which would yield a rule that reuses all nodes in X_N.

In other words, if $X_N = \{x_0, \ldots, x_N\}$ and $\widehat{X}_{N+1} = \{\widehat{x}_0, \ldots, \widehat{x}_{N+1}\}$, for each node $x_k \in X_N \setminus \widehat{X}_{N+1}$ the set Ω_l as defined by (3.18) is identified such that $\widehat{X}_{N+1} \cup \{x_k\} \setminus \{\widehat{x}_l\}$ is the nodal set of a positive and interpolatory quadrature rule. If there is an x_k such that $\widehat{x}_l \notin X_N$, we set $\widehat{X}_{N+1} \leftarrow \widehat{X}_{N+1} \cup \{x_k\} \setminus \{\widehat{x}_l\}$ and keep repeating this procedure until no such x_k exists anymore. If there are multiple x_k that could possibly be used to trigger a replacement in $\widehat{X}_{N+1}$, the smallest one is selected in the examples presented in this chapter.

The nodes from X_N that cannot be added to $\widehat{X}_{N+1}$ are reconsidered in consecutive iterations and added again if possible. It is difficult to theoretically quantify the number of nodes from X_N that can be "added" this way to $\widehat{X}_{N+1}$, though it is straightforward to see that there exists at least a single $x_k \in X_N$ that can be reused.

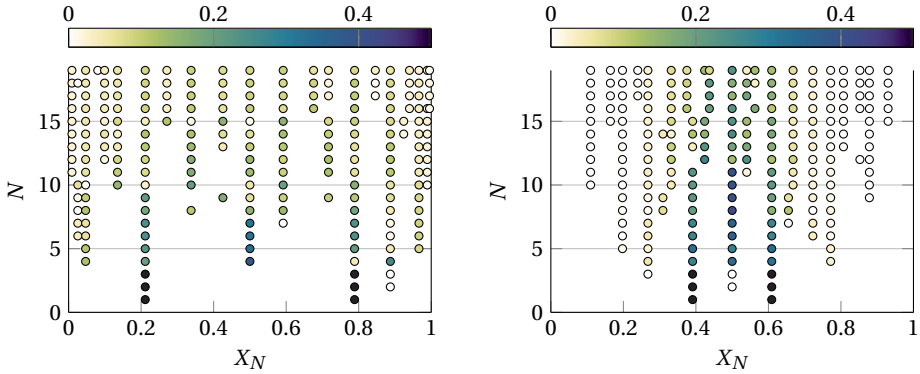
To demonstrate this procedure numerically, let X_1 contain two Gaussian nodes accompanied by the weights W_1. If the uniform distribution is considered, the sequence of quadrature rules up to $N = 19$ (i.e. 20 nodes) only "loses" two nodes in the process (see Figure 3.5a). In other words, two function evaluations used to compute quadrature rule estimations are not being reused by the rule that consists of 20 nodes. This results seems to be somewhat independent of the distribution, since the same sequence of quadrature rules with respect to a Beta(10, 10) distribution only loses three nodes (see Figure 3.5b).

The main advantage of this approach compared to the previously discussed Patterson extension and Leja nodes is that it always yields a quadrature rule with positive weights. Moreover the expressions to compute the nodes contained in the quadrature rule are straightforward. However, the approach does not have a significant advantage over removing nodes from an existing quadrature rule (see Section 3.3), since both approaches need to be initialized with an existing quadrature rules. In other words, both algorithms do not construct quadrature rules from scratch.

3.5. Addition of multiple nodes

In the previous section a counterexample of a positive interpolatory quadrature rule is discussed that cannot be extended by adding a single node. In this section we will therefore study the addition of multiple nodes to a quadrature rule. The problem setting is that of Section 3.2.3: given a positive interpolatory quadrature rule X_N, W_N, determine M as small as possible and nodes X_{N+M} with $X_N \subset X_{N+M}$ such that X_{N+M} forms the nodal set of a positive interpolatory quadrature rule.

The first step is to extend the derivation of Section 3.4.1 for the addition of multiple nodes. Applying Cramer's rule is slightly more complicated in this case and requires a bit more bookkeeping, but the key principles are the same. With the theory that is derived in the upcoming Section 3.5.1 it is not obvious how nodes can be added to the quadrature rule, but it provides geometrical insight in the location of such nodes with respect to the existing nodes. Again we can derive some non-trivial adjustments



(a) Uniform **(b)** Beta(10, 10)

Figure 3.5: Partially nested, positive, and interpolatory quadrature rules constructed using sequences of Gaussian quadrature rules. The colors of the nodes indicate their weight.

one can apply to a quadrature rule. These are discussed in Section 3.5.2. Similar to the case of a single node, there is a tight relation with the Patterson extension. In this case, the Patterson extension for general M is recovered. This is discussed in Section 3.5.3, including some examples of nested quadrature rules obtained with the theory derived in this section.

3.5.1. Positive weight criterion

The idea is similar to the derivation of the addition of single node. Let X_N be the initial nodal set and let M be given. The goal is to determine X_{N+M} with $X_N \subset X_{N+M}$ such that it forms a positive interpolatory quadrature rule.

Let $w_k^{(N)}$ for $k = 0, \ldots, N$ be the weights of W_N and likewise let $w_k^{(N+M)}$ be the (unknown) weights of W_{N+M}. Then there exists a vector $\mathbf{c} = (c_0, \ldots, c_N, c_{N+1}, \ldots, c_{N+M})^{\mathrm{T}}$ such that $w_k^{(N+M)} = w_k^{(N)} + c_k$. The goal is to construct $\mathbf{c}$ such that the obtained rule is interpolatory and positive.

With a similar reasoning as before it is straightforward to observe that the following should hold for such a vector to ensure that the obtained quadrature rule is interpolatory:

$$\sum_{k=0}^{N+M} x_k^j c_k = 0, \text{ for } j = 0, \ldots, N,$$

and

$$\sum_{k=0}^{N+M} x_k^j c_k = \varepsilon_j, \text{ for } j = N+1, \ldots, N+M,$$

where ε_j is as previously introduced, i.e. $\varepsilon_j := \mu_j - \sum_{k=0}^{N} x_k^j w_k^{(N)}$. This can be written in

the form of a linear system as follows:

$$\begin{pmatrix} x_0^0 & \cdots & x_N^0 & x_{N+1}^0 & \cdots & x_{N+M}^0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_0^N & \cdots & x_N^N & x_{N+1}^N & \cdots & x_{N+M}^N \\ x_0^{N+1} & \cdots & x_N^{N+1} & x_{N+1}^{N+1} & \cdots & x_{N+M}^{N+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_0^{N+M} & \cdots & x_N^{N+M} & x_{N+1}^{N+M} & \cdots & x_{N+M}^{N+M} \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_N \\ c_{N+1} \\ \vdots \\ c_{N+M} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \varepsilon_{N+1} \\ \vdots \\ \varepsilon_{N+M} \end{pmatrix}.$$

Applying Cramer's rule to this system requires more bookkeeping, as the right-hand side contains multiple non-zero entries. Let $\boldsymbol{\varepsilon} = (0, \ldots, 0, \varepsilon_{N+1}, \ldots, \varepsilon_{N+M})^{\mathrm{T}}$, then Cramer's rule prescribes

$$c_k = \frac{\det V_{(k)}(X_{N+M})}{\det V(X_{N+M})},$$

where $V_{(k)}(X_{N+M})$ is equal to $V(X_{N+M})$ with the k-th column (indexed from 0) replaced by $\boldsymbol{\varepsilon}$. The numerator can be further expanded as follows:

$$\begin{aligned} \det V_{(k)}(X_{N+M}) &= \sum_{j=N+1}^{N+M} (-1)^{(j+1)+(k+1)} \varepsilon_j \det V_{(j,k)}(X_{N+M}) \\ &= \sum_{j=N+1}^{N+M} (-1)^{j+k} \varepsilon_j \det V_{(j,k)}(X_{N+M}), \end{aligned}$$

where $V_{(j,k)}(X_{N+M})$ is the (j,k)-minor of $V(X_{N+M})$ (i.e. the matrix without its j-th row and k-th column, where both indices start at 0). Hence for c_k the following expression is obtained:

$$\begin{aligned} c_k &= \sum_{j=N+1}^{N+M} (-1)^{j+k} \varepsilon_j \frac{\det V_{(j,k)}(X_{N+M})}{\det V(X_{N+M})} \\ &= \sum_{j=N+1}^{N+M} (-1)^{j+k} \varepsilon_j \frac{\det V_{(j,k)}(X_{N+M})}{\det V_{(N+M,k)}(X_{N+M})} \Bigg/ \left(\prod_{0 \le i < k} (x_k - x_i) \prod_{k < j \le N+M} (x_j - x_k) \right) \\ &= \sum_{j=N+1}^{N+M} (-1)^{j} \varepsilon_j \frac{\det V_{(j,k)}(X_{N+M})}{\det V_{(N+M,k)}(X_{N+M})} \Bigg/ \left(\prod_{\substack{j=0 \\ j \neq k}}^{N+M} (x_j - x_k) \right) \\ &= \sum_{j=N+1}^{N+M} (-1)^{N+M-j} \varepsilon_j \frac{\det V_{(j,k)}(X_{N+M})}{\det V_{(N+M,k)}(X_{N+M})} \Bigg/ \left(\prod_{\substack{j=0 \\ j \neq k}}^{N+M} (x_k - x_j) \right). \end{aligned}$$

The same derivation is commonly used to derive the determinant of a Vandermonde matrix [64, 118], and it is well-known that the ratio of determinants obtained in this expression is an elementary symmetric polynomial. The k-th elementary symmetric

polynomial is generally defined as the sum of all monomial permutations of length k, that is as follows:

$$\sigma_k(x_0,\dots,x_N) = \sum_{0\le i_1<\cdots<i_k\le N} x_{i_1}\cdots x_{i_k}.$$

The elementary symmetric polynomials are only defined for $k \le N+1$ and by convention $\sigma_0 \equiv 1$. Concluding, the following expression is obtained for c_k (with $k = 0,\dots,N+M$):

$$c_k = \left(\sum_{j=N+1}^{N+M} (-1)^{N+M-j}\varepsilon_j \sigma_{N+M-j}(X_{N+M}\setminus\{x_k\})\right) \Bigg/ \left(\prod_{\substack{j=0\\ j\neq k}}^{N+M} (x_k - x_j)\right).$$

Here, σ_k is the k-th elementary symmetric polynomial as defined above. With a little abuse of notation, we used:

$$\begin{aligned}\sigma_{N+M-j}(X_{N+M}\setminus\{x_k\}) &:= \sigma_{N+M-j}(x_0,\dots,x_{k-1},0,x_{k+1},\dots,x_{N+M})\\ &= \sigma_{N+M-j}(x_0,\dots,x_{k-1},x_{k+1},\dots,x_{N+M}).\end{aligned}$$

We are now in a position to formulate a theorem in similar form as Theorem 3.3, but then for multiple nodes.

Theorem 3.4 (Addition of multiple nodes). *Let X_N, W_N form an interpolatory quadrature rule. Then $X_{N+M} = X_N \cup \{x_{N+1},\dots,x_{N+M}\}$ forms the nodal set of a positive interpolatory quadrature rule if and only if*

$$-\left(\sum_{j=N+1}^{N+M} (-1)^{N+M-j}\varepsilon_j \sigma_{N+M-j}(X_{N+M}\setminus\{x_k\})\right) \Bigg/ \left(\prod_{\substack{j=0\\ j\neq k}}^{N+M} (x_k - x_j)\right) \le w_k^{(N)},$$

for $k = 0,\dots,N+M$.

For $M = 1$, we have that the summation only incorporates $j = N+1$, hence we have

$$\sigma_{N+M-j}(X_{N+M}\setminus\{x_k\}) = \sigma_0(X_{N+M}\setminus\{x_k\}) = 1,$$

so Theorem 3.3 is recovered, hence Theorem 3.4 is indeed a strict generalization of Theorem 3.3.

3.5.2. Quadrature rule adjustments

Theorem 3.4 presents a necessary and sufficient condition for a quadrature rule extended with M nodes to have positive weights. Contrary to the addition of a single node, it cannot be used directly to determine possible nodes that can be added to the quadrature

rule. This can be seen by rewriting it in a similar form as (3.14), i.e. for $k = 0, \ldots, N+M$:

$$\begin{aligned} w_k^{(N)} \prod_{\substack{j=0 \\ j\neq k}}^{N+M} (x_k - x_j) &\geq -\sum_{j=N+1}^{N+M} (-1)^{N+M-j} \varepsilon_j \sigma_{N+M-j}(X_{N+M} \setminus \{x_k\}) \quad \text{if} \prod_{\substack{j=0 \\ j\neq k}}^{N+M} (x_k - x_j) \geq 0, \\ w_k^{(N)} \prod_{\substack{j=0 \\ j\neq k}}^{N+M} (x_k - x_j) &\leq -\sum_{j=N+1}^{N+M} (-1)^{N+M-j} \varepsilon_j \sigma_{N+M-j}(X_{N+M} \setminus \{x_k\}) \quad \text{if} \prod_{\substack{j=0 \\ j\neq k}}^{N+M} (x_k - x_j) \leq 0. \end{aligned} \tag{3.20}$$

Notice that, if $x_{N+1}, \ldots, x_{N+M}$ are unknowns, an M-variate system of $N+M+1$ polynomial inequalities is obtained. In general these systems are very difficult to solve, so we do not directly pursue a solution of the system above. Nonetheless, the system still provides a geometrical interpretation about where solutions reside, similar to the case of single node addition (though less intuitive). Based on these geometrical insights, procedures to replace nodes and to add nodes, which extend those explained previously, can be derived.

Geometry of nodal addition

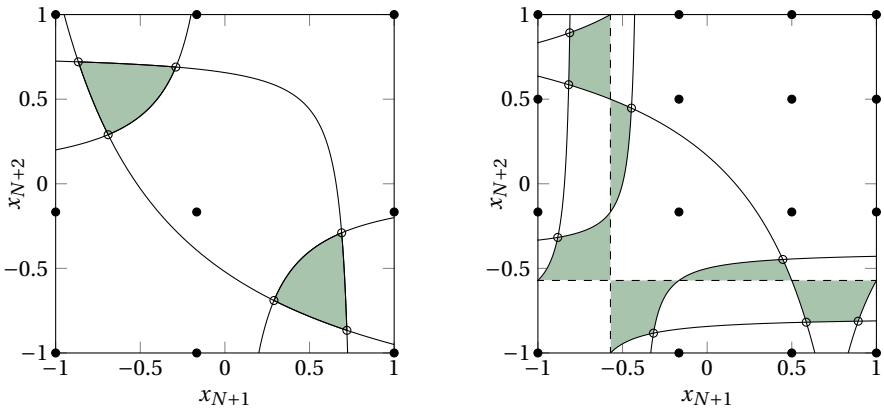
The type of the inequalities (3.20) (i.e. "greater than" versus "less than") does not change between two nodes and if this type is fixed, the system consists of polynomial inequalities. Hence the region where M nodes can be added is described by a continuous boundary, bounded by the polynomial inequalities of (3.20), consisting of lines, surfaces, or "hypersurfaces" through the nodes.

If one of the right-hand sides of (3.20) changes sign, there is an addition of M nodes such that the inequality forms an equality for a specific k. In such cases, there is an addition such that one of the nodes obtains a weight equal to zero. This is equivalent to the replacement operation discussed in Section 3.4.2, where a single node is added in order to set the weights of another node equal to zero.

It is difficult to visualize the addition of M nodes in a similar way as we visualized the addition of one node, as there are M nodes $x_{N+1}, \ldots, x_{N+M}$ and M quadrature rule errors $\varepsilon_{N+1}, \ldots, \varepsilon_{N+M}$. Plotting the errors with respect to the nodes (as in Figure 3.4) is therefore not viable, as this is a plot from $\mathbb{R}^M$ to $\mathbb{R}^M$.

On the other hand, if the distribution $\rho(x)$ is fixed beforehand, the values of ε_{N+1}, $\ldots, \varepsilon_{N+M}$ are known and contour plots of the regions encompassing all M nodes that can be added can be made (provided that M is small enough). For example, let $\rho \equiv 1/2$ with $\Omega = [-1, 1]$. In Figure 3.6 lines are depicted where the inequalities from (3.20) are equalities. The shaded area depicts regions where all inequalities are valid, i.e. any coordinate (x_{N+1}, x_{N+2}) in the shaded region can be added to the respective quadrature rule in order to obtain a positive interpolatory rule. The figure is obviously symmetric around $x_{N+1} = x_{N+2}$, as the order of addition (i.e. first adding x_{N+1} and then x_{N+2} or vice versa) yields equivalent quadrature rules. Selecting a coordinate (x_{N+1}, x_{N+2}) on one of the boundaries results in one weight equal to zero. Adding the coordinates on the corners, depicted by the open circles (i.e. "the boundary of the boundary"), results in two weights equal to zero.

The dashed lines indicate where the inequalities (3.20) with $k = N+1$ and $k = N+2$ change sign. If this happens, one of the new nodes x_{N+1} or x_{N+2} has weight equal to



(a) $X_N = \{-1, -1/6, 1\}$ **(b)** $X_N = \{-1, -1/6, 1/2, 1\}$

Figure 3.6: Two examples of addition of two nodes to a quadrature rule. Choosing the two nodes in a shaded area yields positive weights. Choosing the two nodes on the open circles yields two weights equal to zero and positive weights. Dashed lines correspond to a zero weight for x_{N+1} or x_{N+2}.

zero. This line forms everywhere a boundary of the shaded area: the node with weight equal to zero can be replaced by any other node, while still resulting into an interpolatory quadrature rule with positive weights. This situation is equivalent to adding a single node x_{N+1} to the quadrature rule, but gaining two degrees, as discussed before.

The addition and replacement of multiple nodes follow readily from these insights. Notice that if any coordinate $(x_{N+1}, \ldots, x_{N+M})$ is known, the replacement for $M = 1$ can be used to reach any other coordinate $(x_{N+1}, \ldots, x_{N+M})$ in the same region (shaded in Figure 3.6). Hence if all corners of those regions are determined (depicted as open circles in Figure 3.6), the full region can be explored straightforwardly. As these corners form a replacement of nodes, we start by discussing replacement of M nodes. Moreover, it will be shown that these corners are a Patterson extension. Based on the algorithm to determine all these corners, addition of M nodes follows straightforwardly.

Replacement of multiple nodes

Let X_N, W_N be an interpolatory quadrature rule and let indices $k_1, \ldots, k_M$ be given such that $0 \leq k_i \leq N$ and $k_i \neq k_j$ for $i \neq j$. In this section the goal is to determine the interpolatory quadrature rule X_{N+M}, W_{N+M} such that $w_{k_i}^{(N+M)} = 0$ for all k_i. Notice that this is equivalent to replacing the nodes $x_{k_1}, \ldots, x_{k_M}$ in the quadrature rule X_N by the nodes $x_{N+1}, \ldots, x_{N+M}$. The nodes with this property are the intersections of the polynomials of (3.20) and they are depicted as open circles in Figure 3.6. Moreover, they describe the boundary of the set of nodes that can be added to the quadrature rule.

The desired nodes $x_{N+1}, \ldots, x_{N+M}$ can be determined by calculating the Patterson extension of the interpolatory quadrature rule with the nodes $X_N \setminus \{x_{k_1}, \ldots, x_{k_M}\}$, for which efficient techniques exist [102, 103, 141]. Such techniques require that M must be

known a priori and they do not provide a simple geometrical interpretation. Therefore we proceed by embedding the Patterson extension in the framework discussed here. This yields a new alternative algorithm to determine these nodes, though the algorithm is mostly of interest due to its geometrical interpretation.

We start by solving a slightly easier problem. Assume $\varepsilon_{N+1} = \cdots = \varepsilon_{N+M-1} = 0$ and $\varepsilon_{N+M} \neq 0$. Notice that, if ε_{N+M} is neglected, any addition of $M-1$ nodes yields a valid quadrature rule (as these nodes have zero weight). Geometrically, a fully shaded figure (if drawn as Figure 3.6) is obtained. This can be exploited to determine the desired nodes, as only the value of ε_{N+M} imposes a condition on the nodes $x_{N+1}, \ldots, x_{N+M}$.

The nodes that yield $w_{k_1}^{(N+M)} = \cdots = w_{k_M}^{(N+M)} = 0$ can be found by applying Theorem 3.4 with $c_{k_i} = -w_{k_i}^{(N)}$ for all i or by consecutively applying Theorem 3.3. In both cases, the following is obtained:

$$\varepsilon_{N+M} = -w_{k_i}^{(N)} \left(\prod_{\substack{j=0 \\ j \neq k_i}}^{N} (x_{k_i} - x_j) \right) \left(\prod_{j=N+1}^{N+M} (x_{k_i} - x_j) \right), \text{ for } i = 1, \ldots, M. \tag{3.21}$$

In principle this system of polynomial equalities is difficult to solve, but it has a certain structure that can be exploited. To see this, let $\widehat{L}_M(x)$ be the nodal polynomial of the nodes $x_{N+1}, \ldots, x_{N+M}$:

$$\widehat{L}_M(x) = \prod_{j=N+1}^{N+M} (x - x_j),$$

which translates the system above to

$$\varepsilon_{N+M} = -w_{k_i}^{(N)} \left(\prod_{\substack{j=0 \\ j \neq k_i}}^{N} (x_{k_i} - x_j) \right) \widehat{L}_M(x_{k_i}), \text{ for } i = 1, \ldots, M. \tag{3.22}$$

If the nodal polynomial $\widehat{L}_M$ is known, its roots equal $x_{N+1}, \ldots, x_{N+M}$. The nodal polynomial has degree M and it is known that its leading order coefficient equals 1. Therefore it is useful to introduce the polynomial $q_M(x) := \widehat{L}_M(x) - x^M$, which has degree $M-1$. Then (3.22) can be rewritten as follows:

$$q_M(x_{k_i}) = \widehat{L}_M(x_{k_i}) - x_{k_i}^M = -\,\varepsilon_{N+M} \Bigg/ \left(w_{k_i}^{(N)} \prod_{\substack{j=0 \\ j \neq k_i}}^{N} (x_{k_i} - x_j) \right) - x_{k_i}^M, \text{ for } i = 1, \ldots, M.$$

These are M values of a polynomial of degree $M-1$, which is a well-known interpolation problem and can be solved with various well-known methods (such as barycentric interpolation [7]). If q_M is determined, the roots of the polynomial $\widehat{L}_M(x) = q_M(x) + x^M$ are the nodes $x_{N+1}, \ldots, x_{N+M}$. By construction these nodes are such that $w_{k_i}^{(N+M)} = 0$ for $i = 1, \ldots, M$.

Even though assuming $\varepsilon_{N+1} = \cdots = \varepsilon_{N+M-1} = 0$ is not realistic in practical cases, this procedure can readily be extended to the general case. For this we reuse the replacement

Algorithm 3.1: Determining X_{N+M} with zero weights

Input: Interpolatory quadrature rule X_N, W_N, indices $k_1, \ldots, k_M$.
Output: Interpolatory quadrature rule X_{N+M}, W_{N+M} such that $w_{k_i}^{(N+M)} = 0$ for all i.

$m \leftarrow 1$
for $k = k_1, \ldots, k_M$ **do**
  Determine $\widehat{L}_m$ such that $\widehat{L}_m(x) = x^m + q_m(x)$ (see text) and

$$\varepsilon_{N+m} = -w_l \widehat{L}_m(x) \prod_{\substack{j=0 \\ j \neq l}}^{N+m} (x_l - x_j), \text{ for both } l = k \text{ and } l = N+1, \ldots, N+m-1$$

  Let $r_1, \ldots, r_m$ be the roots of $\widehat{L}_m$, i.e. $\widehat{L}_m(r_k) = 0$
  $X_{N+m} \leftarrow X_N \cup \{r_1, \ldots, r_m\}$ and determine W_{N+m}
  $m \leftarrow m + 1$
end for
Return X_{N+M}, W_{N+M}

step. If $\varepsilon_{N+1} \neq 0$, then a single node is added to the quadrature rule such that $w_{k_1}^{(N+1)} = 0$. This is equivalent to a replacement of a single node. Then the obtained quadrature rule $X_{N+1} \setminus \{x_{k_1}\}$ has $\varepsilon_{N+1} = 0$. By applying the procedure discussed above to these $N+1$ nodes, the nodes x_{N+2} and x_{N+3} can be determined such that $w_{k_2}^{(N+2)} = 0$ and $w_{N+1}^{(N+2)} = 0$, i.e. we enforce that the weight of x_{k_2} is zero and the weight of the previously added node becomes zero. The obtained rule has $N+3$ nodes, where two nodes have weight equal to zero. This is again a replacement, but here *two* nodes get weight equal to zero. Those nodes are removed to reobtain a quadrature rule of $N+1$ nodes and this process is repeated iteratively until X_{N+M} is obtained. The obtained rule can be interpreted as a replacement of M nodes, and yields the open circles from Figure 3.6. It is an iterative description: a replacement of M nodes is determined using a replacement of $M-1$ nodes. Geometrically, the algorithm iterates over the dimension of the figure and iteratively determines a set of nodes that can be used as a replacement.

The obtained nodes form by definition a Patterson extension of the nodal set $X_N \setminus \{x_{k_1}, \ldots, x_{k_M}\}$, since it holds that $X_N \setminus \{x_{k_1}, \ldots, x_{k_M}\} \cup \{x_{N+1}, \ldots, x_{N+M}\}$ has degree $N+M$. The existence of such a Patterson extension is directly coupled to the existence of M nodes that can possibly be added to X_N in the hope of obtaining an interpolatory quadrature rule with positive weights: if M nodes can be added to the quadrature rule, the Patterson extension has positive weights, since it forms the boundary of the set that describes all additions. Moreover, if all Patterson extensions of all sets $X_N \setminus \{x_{k_1}, \ldots, x_{k_M}\}$ for any sequences $(k_1, \ldots, k_M)$ have negative weights or are not real-valued, no addition of M nodes exists.

The approach is outlined in Algorithm 3.1. By iterating over all possible sorted sequences $(k_1, \ldots, k_M)$, this procedure can be used straightforwardly to verify whether there exist M nodes that can be added to a given quadrature rule (though this is a costly procedure).

There are two special cases that are (for sake of simplicity) not incorporated in Algorithm 3.1. Firstly, if $w_k^{(N+m)} = 0$ at the start of an iteration, the polynomial $\widehat{L}_M(x)$ is not well defined. This can be incorporated by selecting any non-zero $w_{k_i}^{(N+m)}$ at the start of the iteration. If no such $w_{k_i}^{(N+m)}$ exists, then all these weights are zero, which is the primary goal of the algorithm. Secondly, if $r_k \in X_N$ or $\varepsilon_{N+m} = 0$, a quadrature rule is obtained that has higher degree than its number of nodes. This can be incorporated by combining all double nodes in X_N and likewise adding the respective weights and by skipping any iteration that has $\varepsilon_{N+m} = 0$.

Addition of multiple nodes

By combining the quadrature rule replacement of Section 3.4.2 (for $M = 1$) and the replacement of the previous section (for $M > 1$), we obtained a naive algorithm to firstly determine M as small as possible such that there exists a positive interpolatory quadrature rule X_{N+M} and secondly to explore *all* such M nodes (i.e. the shaded areas of Figure 3.6).

Determining the number of nodes M that can be added to an interpolatory quadrature rule can straightforwardly be done by solving (3.21) for each sequence of $k_1, \ldots, k_M$ with $k_1 < \cdots < k_M$. This gives all locations where M nodes have zero weight. If at any of these locations all nodes have non-negative weight, then M nodes can be added to the rule. Otherwise, M is increased and the process is repeated.

Often the value of M is unknown a priori. Besides determining the M nodes that can be added, the goal is also to determine M as small as possible (this is also how we formulated the problem originally in Section 3.2.3). Algorithm 3.1 can be used to efficiently determine M, as results from previous iterations can be reused. To see this, suppose a quadrature rule is given and by applying Algorithm 3.1 it is known that no addition of at most $M-1$ nodes exist. Then during these calculations, all sequences of nodes have been determined that make $M-1$ weights zero. By initializing Algorithm 3.1 with these sequences, only the last iteration of the loop is necessary, which significantly reduces the computational expense.

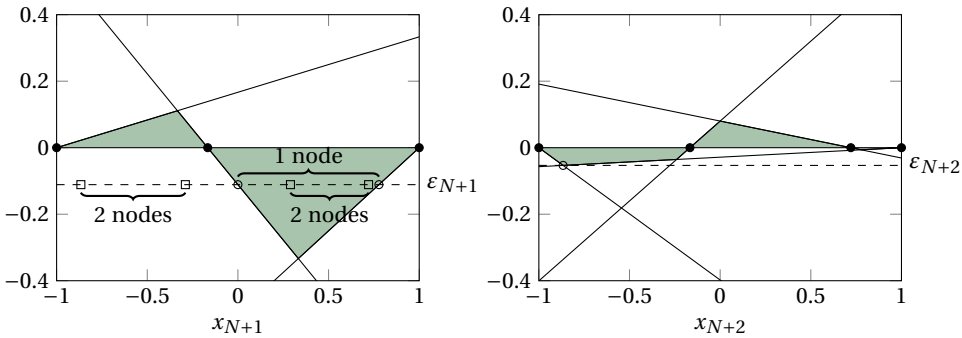
It is required to repeatedly determine large numbers of polynomial roots in this algorithm. This is nearly impossible to do symbolically, except for some special cases (e.g. $M \leq 3$ or symmetric quadrature rules). Moreover determining the roots numerically can result in quick aggregation of numerical errors. We use variable precision arithmetic, i.e. determine the roots with a large number of significant digits.

For large N this is a costly algorithm, as the number of sorted sequences of length M equals

$$\#(k_1, \ldots, k_M) = \binom{N+1+M}{M},$$

which grows fast for large N. Therefore using this algorithm to compute all removals is slower than using existing techniques to compute the Patterson extension, albeit that it is able to reuse all additions of $M-1$ nodes to compute all additions of M nodes.

If all sets of M nodes have been determined that can be added to the quadrature rule, the techniques from Section 3.4.2 can be used to fully explore all nodes that can be



(a) Addition of two nodes

(b) Addition of one node

Figure 3.7: The addition of 2 nodes to the interpolatory quadrature rule with the nodes $X_N = \{-1, -1/6, 1\}$. The right quadrature rule is obtained by adding the rightmost highlighted node of the left figure (i.e. "the rightmost square").

added to the rule. This requires solving linear equalities, which can be done fast and accurately.

Notice that the possibility of adding M nodes to the quadrature rule does not guarantee the possibility of adding $M+1$ nodes to the quadrature rule. This can be observed by revisiting the quadrature rule example from the previous section, i.e.

$$X_N = \left\{-1, -\frac{1}{6}, 1\right\}, W_N = \left\{\frac{1}{10}, \frac{24}{35}, \frac{3}{14}\right\}.$$

In Figure 3.7a regions are depicted where a single node can be added (similar to Figure 3.4a) and regions where, upon adding a node from that region, another node can be added (this is the projection of Figure 3.6a). The addition of the rightmost node with the latter property is depicted in Figure 3.7b, demonstrating that there is a single node that can be added and that this is indeed a limiting case.

Notice that the intervals where a single node and where two nodes can be added are independent of each other. There exist pairs of nodes x_{N+1}, x_{N+2} firstly such that both W_{N+1} and W_{N+2} are all positive (in the right interval surrounded by squares), secondly such that W_{N+1} is positive, but W_{N+2} is not (the right interval surrounded by circles, outside the interval surrounded by squares), thirdly such that W_{N+1} is not positive, but W_{N+2} is (the left interval surrounded by squares), and finally such that both W_{N+1} and W_{N+2} are always negative (outside all intervals).

3.5.3. Constructing quadrature rules

Similar to the case of addition of a single node, the Patterson extension is obtained for specific choices of nodes that are added to the rule. In fact, the nodes determined with Algorithm 3.1 form a Patterson extension of a quadrature rule with a smaller number of nodes. As the Gaussian quadrature rule is a special case of the Patterson extension, this rule also follows from the framework discussed in this chapter.

By repeatedly applying Algorithm 3.1, a sequence of nested quadrature rules can be determined. Even though constructing quadrature rules is not the primary focus of this chapter, these rules are presented here.

Patterson extension

The boundary of the set that describes all possible additions is spanned by the Patterson extension (the open circles in Figure 3.4a and Figure 3.6a). These nodes have the property that, upon adding them to the quadrature rule, a rule of degree $N+M$ is obtained with M weights equal to zero. This is equivalent to the Patterson extension of the quadrature rule *without* those M nodes with zero weight. For $M=1$, this was demonstrated in Section 3.4.3.

For general M, the Patterson extension can be deduced mathematically as follows. Let X_N, W_N be a quadrature rule and, as before, let $x_{N+1},\dots,x_{N+M}$ be such that the following nodes form a quadrature rule of degree $N+M$:

$$X_N \cup \{x_{N+1},\dots,x_{N+M}\} \setminus \{x_{k_1},\dots,x_{k_M}\}. \tag{3.23}$$

Furthermore, let X_{N-M} be the nodes of an interpolatory quadrature rule of degree $N-M$, defined as follows:

$$X_{N-M} = X_N \setminus \{x_{k_1},\dots,x_{k_M}\}.$$

Upon adding $\{x_{N+1},\dots,x_{N+M}\}$ to X_{N-M}, the nodes from (3.23) are obtained, that have degree $N+M$. Hence M nodes are added to an interpolatory rule of degree $N-M$ and the obtained degree is $N+M$, which is by definition a Patterson extension. Notice that the obtained quadrature rule is interpolatory, but not necessarily positive.

The Gaussian quadrature rule can be deduced as special case from Algorithm 3.1. To see this, suppose $M=N+1$, which is the number of nodes of the rule under consideration. In that case, there is only a single sequence of $k_1,\dots,k_M$, defined as follows up to a permutation:

$$k_j = j-1, \text{ for } j=1,\dots,N+1.$$

By applying Algorithm 3.1, the nodes from (3.23) are obtained with $M=N+1$, which are:

$$X_N \cup \{x_{N+1},\dots,x_{2N+1}\} \setminus \{x_0,\dots,x_N\} = \{x_{N+1},\dots,x_{2N+1}\}.$$

Hence the $N+1$ nodes $x_{N+1},\dots,x_{2N+1}$ form a quadrature rule of degree $2N+1$, which is by definition the Gaussian quadrature rule. In other words, when adding a Gaussian quadrature rule to an existing quadrature rule and setting all existing weights to zero a valid addition is obtained.

To demonstrate where these rules occur in our framework, reconsider the interpolatory quadrature rule with the nodes $X_N = \{-1,-1/6,1\}$. In Section 3.4.3 three different Patterson extensions related to this quadrature rule were discussed: $\{-1/3,1\}$, $\{-1/6,2\}$, or $\{-1,1/3\}$. All these rules are Patterson extensions (of smaller quadrature rules) with $M=1$. To obtain a Patterson extension with $M=2$ and subsequently a Gaussian quadrature rule, consider Algorithm 3.1 using $\{k_1,k_2,k_3\} = \{0,1,2\}$. The algorithm proceeds as follows:

1. In the first iteration, it follows that $\widehat{L}_1(x) = x+5/3$ and therefore the following

quadrature rule is obtained:

$$X_{N+1} = \left\{-1, -\frac{1}{6}, 1, -\frac{5}{3}\right\},$$
$$W_{N+1} = \left\{0, \frac{16}{21}, \frac{11}{56}, \frac{1}{24}\right\}.$$

Notice that the node $x_{N+1} = -5/3$ was obtained in Section 3.4.2, where we discussed that after adding this node one obtains $w_0^{(3)} = 0$.

2. In the second iteration, it follows that $\widehat{L}_2(x) = x^2 + 2/5x - 1/5$. Here, the Patterson extension with $M = 2$ of the quadrature rule with "nodes" $\{1\}$ is obtained. Hence the following rule is obtained (notice that the node $-5/3$ is removed):

$$X_{N+2} = \left\{-1, -\frac{1}{6}, 1, \frac{1}{5}\left(-1-\sqrt{6}\right), \frac{1}{5}\left(-1+\sqrt{6}\right)\right\},$$
$$W_{N+2} = \left\{0, 0, \frac{1}{9}, \frac{1}{36}\left(16+\sqrt{6}\right), \frac{1}{36}\left(16-\sqrt{6}\right)\right\}.$$

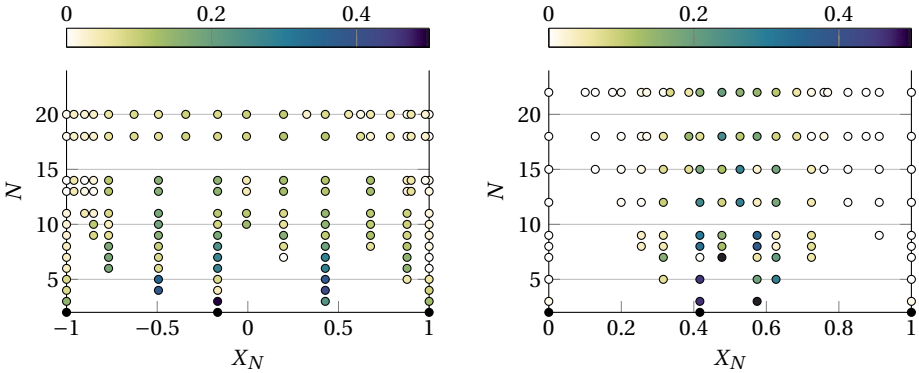
3. In the third iteration, it follows that $\widehat{L}_3(x) = x^3 - 3/5x$, whose roots are the Gaussian quadrature rule or, equivalently, the Patterson extension with $M = 3$ of the empty quadrature rule:

$$X_{N+3} = \left\{-1, -\frac{1}{6}, 1, -\frac{1}{5}\sqrt{15}, 0, \frac{1}{5}\sqrt{15}\right\},$$
$$W_{N+3} = \left\{0, 0, 0, \frac{5}{18}, \frac{4}{9}, \frac{5}{18}\right\}.$$

In this specific example it is possible to determine all nodes symbolically, but for larger values of M this is generally not possible.

Considering the nodes in a different order results in different intermediate Patterson extensions, but obviously the Gaussian quadrature rule is the rule that is finally obtained. These steps also demonstrate the possibility to store intermediate results: only the nodes of step 2 are necessary to deduce the nodes of step 3.

Specialized algorithms exist for specific distributions and specific values of N and M to construct Gaussian, Gauss–Kronrod, and Gauss–Patterson quadrature rules [74, 103], but it remains a challenging topic to determine the Patterson extension for general non-Gaussian quadrature rules. The algorithm presented in this chapter is not an alternative for these existing algorithms, but embeds the Patterson extension in the discussed framework and can be used to determine *all* M nodes that can be added to a quadrature rule. If an efficient procedure to determine large numbers of Patterson extensions is available, it can be readily used to determine whether an extension for a specific M exists. By consecutively replacing the new nodes (see Section 3.4.2) all M nodes that can be added can be found.



(a) Uniform **(b)** Beta(10, 10)

Figure 3.8: Nested, positive, and interpolatory quadrature rules, initialized with $X_N = \{-1, -1/6, 1\}$ (*left*) or $X_N = \{0, 5/12, 1\}$ (*right*). The colors of the nodes indicate their weight, except for the set of initial nodes (with $N = 2$), which are depicted in black.

Nested, positive, and interpolatory quadrature rule

Algorithm 3.1 provides a straightforward procedure to determine the minimal value of M and the positive interpolatory quadrature rule nodes X_{N+M} such that $X_N \subset X_{N+M}$. The replacement procedure for $M = 1$ of Section 3.4.2 can be used to determine all possible nodes, given M. This is the original goal of the chapter as outlined in Section 3.2.3 and examples of such quadrature rules are depicted in Figure 3.8. Here, each quadrature rule is iteratively extended with a minimal number of nodes, and the nodes that are added are selected randomly from the set containing all M nodes that can be added. There are two main differences with the quadrature rules obtained in Section 3.4.3: the rules obtained in this section are positive and nested, but do add more than one node between two consecutive rules.

Both figures demonstrate that M varies significantly and does not increase monotonically. This is in line with the conclusions drawn in the Section 3.5.2, as shown in Figure 3.7. Moreover for almost all N, the value of M is significantly larger in case the Beta distribution is considered, which is related to the "bad" initial set of nodes for this distribution (these nodes form a quadrature rule with a negative weight). A different initialization would lead to different values of M. We will further study the performance of this quadrature rule in Section 3.6.

3.6. Numerical examples

This chapter is concerned with the construction of quadrature rules with *positive* weights and three new quadrature rules have been introduced: one based on the removal of nodes initiated with an existing quadrature rule, one based on the consecutive replacement of single nodes (possibly resulting in a sequence of rules that is not nested), and one by randomly adding nodes ensuring positive weights. We briefly assess the numer-

ical performance of these quadrature rules by means of the Genz test functions (see Table 3.1). The Genz test functions [66] are functions defined on $\Omega = [0,1]$ constructed specifically to test integration routines. Each function has a specific family attribute that is considered to be challenging for integration routines, that can be enlarged by a shape parameter a and translated by a translation parameter b. These functions are used often in this thesis to assess the accuracy of the proposed integration routines. In this chapter the univariate Genz functions are considered and we restrict ourselves to the uniform distribution, as in this case the exact value of the integral of the Genz functions is known analytically.

We consider the performance of the following six quadrature rules:

1. A quadrature rule that is determined by consecutively adding and replacing nodes originating from a Gaussian quadrature rule (see Figure 3.5a). This rule was discussed in Section 3.4.3 and is a partially nested, positive, and interpolatory quadrature rule. The rule is initialized with the quadrature rule nodes $X_N = \{0, 5/12, 1\}$ (i.e. the nodes from the example as discussed before, translated to $[0,1]$).

2. A quadrature rule that is determined by consecutively randomly adding M nodes to the rule such that the obtained rule is positive. Here M is minimal, i.e. the smallest number of nodes is added for each N (see Figure 3.8a). This rule was discussed in Section 3.5.3 and is a nested, positive, and interpolatory quadrature rule. The rule is initialized in the same way as the quadrature rule of the previous point, i.e. using $X_N = \{0, 5/12, 1\}$.

3. The Clenshaw–Curtis quadrature rule [35], where the nodes X_N are defined explicitly by (2.16). It is well known that these nodes have positive weights if the distribution under consideration is uniform, which is the case. This positive and interpolatory quadrature rule is nested for specific levels, as discussed in Section 2.1.3.

4. The Gaussian quadrature rule [74], where the nodes and weights are defined as the quadrature rule with $N+1$ nodes of degree $2N+1$. This quadrature rule is not nested, so refining the quadrature rule results in a significant number of new function evaluations.

5. Leja nodes [106], as discussed in Section 3.4.3. These nodes do not have positive weights, but form a nested sequence of interpolatory quadrature rules. In other words, it is a nested interpolatory quadrature rule with minimal M and some negative weights.

6. The reduced quadrature rule, which was introduced in Section 3.3. The rule is initiated with a Gauss–Legendre quadrature rule of 30 nodes. Each iteration consists of removing a node such that a sequence of positive, nested, and interpolatory quadrature rules is obtained. There are always two nodes that can be removed from the rule and in this numerical experiment the node that is removed is chosen randomly.

Table 3.1: The test functions from Genz [66], which depend on the shape and translation parameters a and b.

Integrand Family	Attribute
$u_1(x) = \cos(2\pi b + ax)$	Oscillatory
$u_2(x) = \left(a^{-2} + (x-b)^2\right)^{-1}$	Product Peak
$u_3(x) = (1+ax)^{-2}$	Corner Peak
$u_4(x) = \exp\left(-a^2(x-b)^2\right)$	Gaussian
$u_5(x) = \exp(-a\lvert x-b\rvert)$	C^0 function
$u_6(x) = \begin{cases} 0 & \text{if } x > b \\ \exp(ax) & \text{otherwise} \end{cases}$	Discontinuous

The error measure e_N is the absolute integration error, i.e.

$$e_N(u) = |\mathcal{I}u - \mathcal{A}_N u|,$$

where $u = u_g$ with $g = 1, \ldots, 6$, i.e. u is one of the Genz test functions. To obtain meaningful results we select the parameters a and b randomly in the unit interval and repeat the experiment 100 times. This also affects the reduced quadrature rule: each experiment selects the node that is removed randomly and therefore 100 different sequences of nested quadrature rules are obtained. The errors reported here are averaged over the 100 experiments and are therefore denoted by $\overline{e}_N$.

It is instructive to compare the error with the upper bound that follows from the Lebesgue inequality (2.12):

$$e_N(u) \leq 2 \inf_{\varphi \in \Phi_N} \|u - \varphi\|_\infty, \tag{3.24}$$

where we use that $\mu_0 = 1$ in our test cases. This error is determined using the algorithm of Remez [188, Chapter 3], with the implementation from `chebfun` [47]. Convergence results for the uniform distribution $\rho \equiv 1$ in $\Omega = [0,1]$ are gathered in Figure 3.9.

Notice that regardless of the function under consideration all quadrature rule errors remain far under the dashed line, that represents the right-hand side of (3.24). This shows that the bound from this inequality is far from sharp.

The first four Genz functions can be approximated well using polynomials, as they are analytic and have rapidly converging Chebyshev coefficients. The best approximation converges exponentially in these cases, which is also the case for the six quadrature rules under consideration. The quadrature rules determined using the framework of this chapter and the rule based on Leja nodes perform slightly worse than the Clenshaw–Curtis and the Gaussian quadrature rule. This is related to the fact that these rules exploit the structure of the underlying distribution to a large extent (e.g. symmetry and higher-order moments), whereas the rules in this work only optimize for the positivity of the weights. The Gaussian quadrature rule converges with the highest rate, which is related to its high polynomial degree (a rule of $N+1$ nodes has degree $2N+1$). However, the Gaussian rule is not nested, so to refine the estimate of the integral for increasing number of nodes the number of function evaluations increases significantly. If a computationally expensive function is considered, using a nested quadrature rule with fine

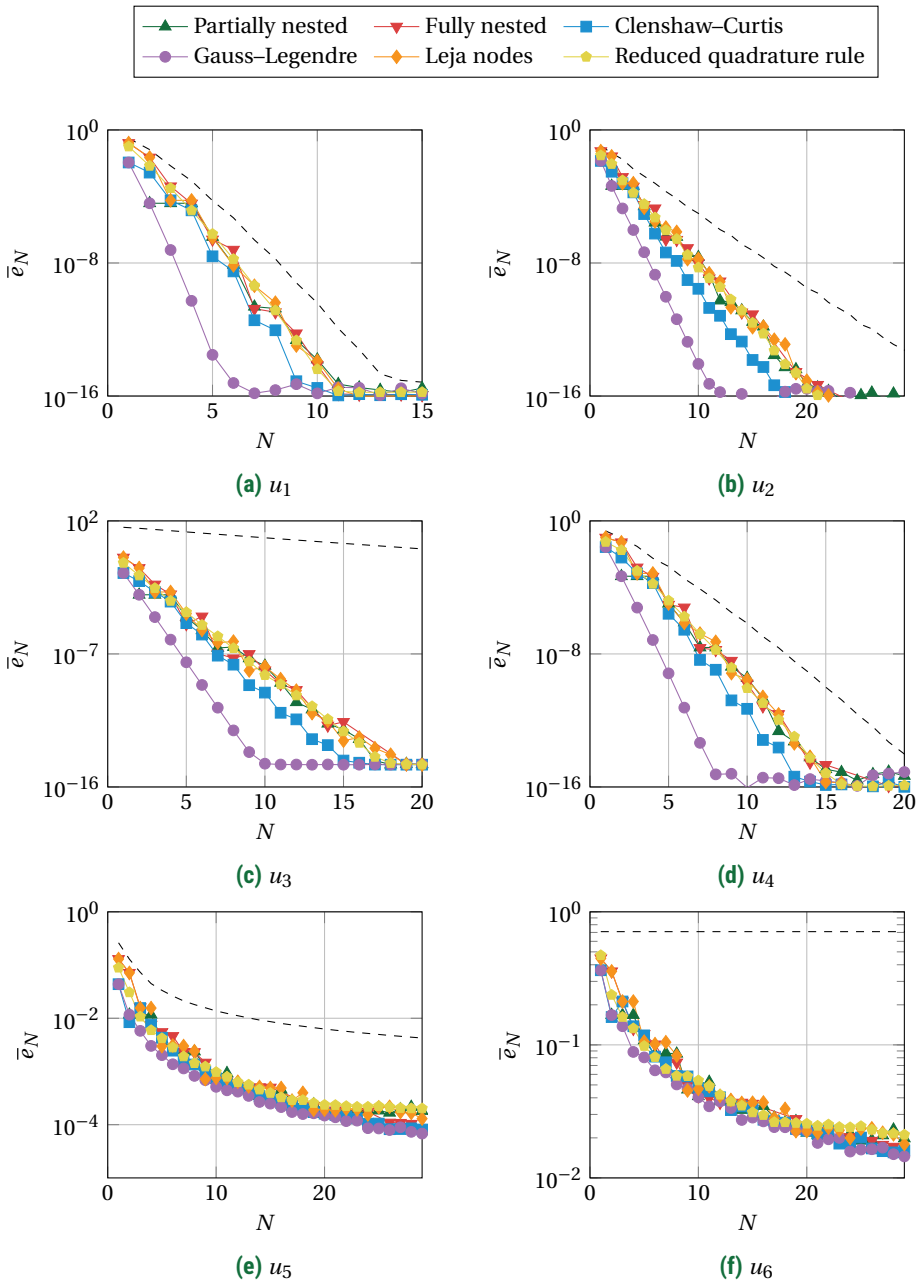


Figure 3.9: Convergence of the Genz test functions using various quadrature rule techniques. The absolute error of the best approximation polynomial (i.e. $\inf_{\varphi\in\Phi_N} \|u-\varphi\|_\infty$) is dashed.

granularity (such as the proposed rules) significantly reduces the cost of refining the quadrature rule estimate.

The fifth Genz test function is not differentiable and can therefore not be approximated well using a polynomial. This can be observed from the best approximation polynomial, that converges with order 1 (so we would expect that $\overline{e}_N \sim 1/N$). In this case the difference between the Gaussian rule and the other rules is significantly smaller, demonstrating that the high polynomial degree of Gaussian rules is less relevant if the integrand is not smooth.

The sixth Genz test function cannot be approximated accurately using a polynomial when considering the ∞-norm, as it is discontinuous. Hence the best approximation error remains constant. However, the approximation of the quadrature rules still converges with order $1/2$. In this case, there is a clear difference between the integration error (that is an averaged error) and the best approximation error (that is a uniform error).

3.7. Conclusion

In this chapter, a novel mathematical framework is presented for the construction of nested, positive, and interpolatory quadrature rules by using a geometrical interpretation. Given an existing quadrature rule, a constructive algorithm has been presented to determine up to two nodes that can be removed from the rule such that the newly obtained rule is again interpolatory and positive. Moreover, necessary and sufficient conditions have been derived for M new nodes to form an interpolatory quadrature rule with positive weights after addition to the quadrature rule. These conditions have been formulated as inequalities, which are explicit if $M = 1$ and implicit if $M > 1$.

The removal of a single node from a quadrature rule is governed by Carathéodory's theorem and there exist at least up to two nodes that can be removed from the quadrature rule such that positivity is preserved. The removal of nodes can be straightforwardly extended to multivariate spaces and does not require any knowledge about the underlying distribution. The latter property will be used extensively in this thesis to construct multivariate quadrature rules (in particular in Chapters 4 and 7).

The addition of a single node can be treated as a special case, which can be solved analytically. The analytical expression can be used to add nodes to and replace nodes within a quadrature rule. The addition of multiple nodes can be determined numerically and a naive algorithm is presented for this purpose.

The replacement of nodes follows from adding nodes such that the weights of existing nodes vanish. Replacement of a single node is always possible and for each node there is a set of nodes that can be used as replacement while preserving positive weights. By determining whether addition of nodes is feasible, the set that encompasses all additions of M nodes can be explored by iteratively replacing nodes.

The well-known Patterson extension of quadrature rules forms a special case of the framework, as it is obtained by constructing the quadrature rules with M weights equal to zero. Also Leja nodes can be interpreted as a heuristic that describes the addition of single nodes. As such, our proposed framework and its geometrical interpretation are well embedded in existing theory on the addition of nodes to quadrature rules. The framework provides various possibilities to construct and change quadrature rules

and three examples have been discussed: one based on the removal of nodes from an existing rule, one based on consecutively adding and replacing one node, and one based on consecutively adding multiple nodes.

There are various options to further extend the framework set out in this chapter. The algorithm to determine whether multiple nodes exist that can be added to the quadrature rule depends on determining many polynomial roots and iterates over all possible sequences of nodes that can become zero. For a large number of nodes this is computationally very costly and therefore warrants the need to derive an efficient algorithm to determine these nodes. Moreover the framework set out in this chapter does not use the relations that exist between consecutive moments of a distribution [166], which can possibly be used to further enhance theory behind the addition of nodes.

4

Constructing quadrature rules on arbitrary sample sets

The framework proposed in the previous chapter describes the addition, replacement, and removal of quadrature nodes in a *univariate* setting. Whereas the removal of nodes can be extended straightforwardly to a more general setting, it is significantly less trivial to generalize the addition and replacement of nodes to a multivariate setting. Moreover the framework requires the exact value of the moments under consideration, which are rarely known if the distribution is inferred from measurement data. These shortcomings are addressed in this chapter and result in a new family of quadrature rules, which we call the *implicit quadrature rule*.

The key idea of the implicit quadrature rule is to interpret a large set of measurements or samples of a distribution as a quadrature rule with constant weights (i.e. it is a Monte Carlo estimation). By repeatedly applying the removal step, as outlined in Section 3.3, an interpolatory quadrature rule with positive weights can be derived. The accuracy of the sample set carries to a certain extent over to the quadrature rule, even though the quadrature rule requires significantly less model evaluations to estimate weighted integrals. The idea of removing nodes from a large set of samples is in theory applicable to any uni- or multivariate setting and does not require an analytical expression of the distribution that generates the samples.

However, in practice it is impossible to consider an arbitrary sized sample set, since accurately computing the null vector that governs the removal of nodes is intractable for very large matrices. To alleviate this, an iterative algorithm is proposed, which is based on the replacement of nodes as outlined in Section 3.4.2. By iteratively considering the samples, instead of considering the sample set as a whole, a multivariate nodal addition

---

This chapter is based on the following article: L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Generating nested quadrature rules with positive weights based on arbitrary sample sets. *To appear in SIAM/ASA Journal on Uncertainty Quantification*, 2018. arXiv: 1809.09842 [math.NA]. An implementation of the algorithms discussed in this chapter is freely available: L. M. M. van den Bos. The implicit quadrature rule. Zenodo, Software, 2019. DOI: 10.5281/zenodo.3234434.

is derived, which consequently results in a mathematical description of the replacement of nodes (by firstly adding a node and secondly removing it). This iterative approach significantly reduces the computational cost of constructing the rule and makes it feasible to compute quadrature rules with large numbers of nodes that incorporate very large numbers of samples. The obtained quadrature rule is therefore applicable to many computationally challenging problems and will in Chapter 5 be used to assess equivalent loads on a turbine.

In this chapter, two variants of the implicit quadrature rule are explained: a non-nested, interpolatory variant and a nested variant which is not necessarily interpolatory. Some numerical results are presented to demonstrate the applicability of the quadrature rule to numerical integration problems.

4.1. Introduction

The problem of uncertainty propagation is considered (see Section 2.1), where the interest is in the effect of uncertainties in model inputs on model predictions. The distribution of the quantity of interest is assessed non-intrusively, i.e. by means of collocation. Problems of this form occur often in engineering applications if boundary or initial conditions are not known precisely. The canonical approach is firstly to identify uncertain input parameters, secondly to define a distribution on these parameters, and finally to determine statistics of the quantity of interest [105, 129, 201]. These statistics are defined as integrals, such as (2.1) introduced in the beginning of this thesis (see page 8), and various techniques exist to approximate these. However, in practice it often occurs that the distribution of the uncertain parameters is known only through a sequence or collection of samples and that the distribution is possibly correlated, e.g. the distribution is inferred through Bayesian analysis or only known by a finite number of measurements. The goal of this chapter is to construct quadrature rules that are accurate for determining integrals when only samples of the distribution are known.

Several approaches exist to tackle problems of this type. As discussed before, in many cases the well-known and straightforward Monte Carlo approach is not applicable due to its low convergence rate of $1/\sqrt{N}$ (with N the number of model evaluations) and instead collocation techniques based on polynomial approximation can be constructed to alleviate this for reasonably small dimensionality. Often these techniques are based on knowledge about the input distribution, for example the moments that were used to derive the quadrature rule framework in Chapter 3. A popular technique to choose evaluation nodes in a multivariate setting is the sparse grid technique [135, 169], which has been extended to a more general, correlated setting (mostly in a Bayesian setting, e.g. [39, 60, 159, 206]), provided that high order statistics of the distribution are known exactly. Other collocation techniques that can be applied to the setting in this chapter are techniques to consider the collocation problem as a minimization problem of an integration error [89, 167], to construct nested rules based on interpolatory Leja sequences [18, 106, 130], or to apply standard quadrature techniques after decorrelation of the distribution [58, 131]. All these approaches provide high order convergence, but require that the input distribution is explicitly known.

On the other hand, procedures that directly construct collocation sequences on samples without using the input distribution directly have seen an increase in popularity,

possibly due to the recent growth of data sets. A recent example is using a clustering approach [54]. Another technique is based on polynomial approximation directly based on data [137] or iteratively with a focus on large data sets [165, 199]. These approaches do not require stringent assumptions on the input distribution, but often do not provide high order convergence.

In this chapter, we propose a novel nested quadrature rule that has positive weights. There are various existing approaches to construct quadrature rules with positive weights. Examples include numerical optimization techniques [89, 96, 156], where oftentimes the nodes and weights are determined by minimization of the quadrature rule error. A different technique that is closely related to the approach discussed in this chapter is subsampling [15, 146, 162, 194], where the quadrature rule is constructed by subsampling from a larger set of nodes. Subsampling has also been used in a randomized setting [199], i.e. by randomly removing nodes from a large tensor grid, or to deduce a proof for Tchakaloff's theorem [5, 42].

The quadrature rule proposed in this chapter is called the *implicit quadrature rule*, because it is constructed using solely samples from the distribution. The nodes of the rule form a subset of the samples and the accompanying weights are obtained by smartly exploiting the null space of the linear system governing the quadrature weights. Using a sample set limits the accuracy of the rule to the accuracy of the sample set, but an arbitrarily sized sample set can be used without additional model evaluations. The computational cost of our proposed algorithm scales (approximately) linearly in the number of samples and for each sample the null space of a Vandermonde matrix has to be determined (whose number of rows equals the number of the nodes of the quadrature rule). The main advantage of using a sample set is that the proposed quadrature rule can be applied to virtually any number of dimensions, basis, space, or distribution without affecting the computational cost of our approach. Moreover it can be extended to obtain a sequence of nested distributions, allowing for refinements that reuse existing (costly) model evaluations.

This chapter is structured as follows. In Section 4.2 the nomenclature and properties of quadrature rules that are specifically relevant for this chapter are discussed. The setting is significantly more general than that of Chapter 3, so some new nomenclature is necessary. In Section 4.3 the implicit quadrature rule is introduced and its mathematical properties are discussed. The accuracy of the quadrature rule is demonstrated by integration of the Genz test functions and by determining the statistical properties of the output of a stochastic partial differential equation modeling the flow over an airfoil. The numerical results of these test cases are discussed in Section 4.4 and conclusions are drawn in Section 4.5.

4.2. Preliminaries

The quantity of interest is modeled as a function $u\colon \Omega \to \mathbb{R}$, as done so far in this thesis, where Ω is a domain in $\mathbb{R}^d$ (with $d = 1, 2, 3, \ldots$). The parameters $\mathbf{x} \in \Omega$ are uncertain and the key problem in this chapter is that their distribution is characterized solely by an arbitrarily large set of samples, denoted by $Y_K := \{\mathbf{y}_0, \ldots, \mathbf{y}_K\} \subset \Omega$ (with $K \in \mathbb{N}$). In other words, the parameters form a multivariate random variable $\mathbf{X}$ with the following discrete

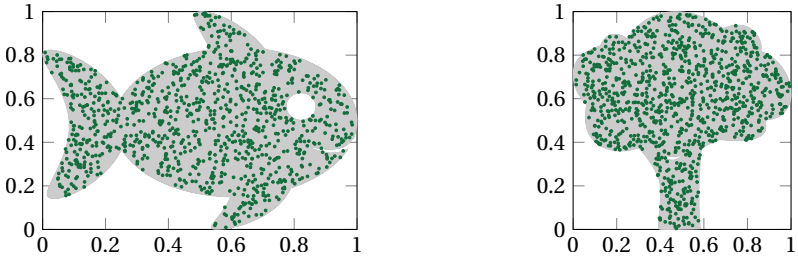


Figure 4.1: The example used throughout this chapter: a uniform distribution restricted to the gray sets. 1000 samples drawn from both distributions are depicted on top of the distributions.

distribution:

$$\rho_K(\mathbf{x}) = \frac{1}{K+1} \sum_{k=0}^{K} \delta(\|\mathbf{x} - \mathbf{y}_k\|),$$

where δ is the Dirac delta function, i.e. $\int_{\mathbb{R}} \delta(x)\,\mathrm{d}x = 1$ and $\delta(x) = 0$ for all $x \neq 0$. Here, $\|\cdot\|$ denotes any norm, since the only necessary property is that $\|\mathbf{a}\| = 0$ if and only if $\mathbf{a} = \mathbf{0}$. The goal is to determine statistical moments of $u(\mathbf{X})$, e.g. to accurately determine

$$\mathcal{I}^{(K)} u := \int_\Omega u(\mathbf{x})\,\rho_K(\mathbf{x})\,\mathrm{d}\mathbf{x} = \frac{1}{K+1} \sum_{k=0}^{K} u(\mathbf{y}_k), \tag{4.1}$$

where higher moments can be determined by replacing $u(\mathbf{x})$ with $u(\mathbf{x})^j$ for given j. Notice that if $\mathbf{y}_k$ are samples drawn from a known (possibly continuous) distribution ρ, (4.1) approximates an integral weighted with this distribution, i.e. the integral from (2.1):

$$\mathcal{I}^{(K)} u = \int_\Omega u(\mathbf{x})\,\rho_K(\mathbf{x})\,\mathrm{d}\mathbf{x} \approx \mathcal{I} u = \int_\Omega u(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{4.2}$$

We will assume throughout this chapter that a large number of samples can be determined fast and efficiently or is provided beforehand. There exist various methods to construct samples from well-known distributions (such as the Gaussian, Beta, and Gamma distribution) [45], from general distributions by means of acceptance rejection approaches, or from unscaled probability density functions by means of Markov chain Monte Carlo methods [80, 122]. Recall that Markov chain Monte Carlo methods were briefly introduced in Section 2.2.2. An example of acceptance rejection sampling that we will use throughout this chapter to visualize the proposed quadrature rule is depicted in Figure 4.1.

If $K+1$ samples $Y_K = \{\mathbf{y}_0, \ldots, \mathbf{y}_K\}$ are given, (4.1) could naively be evaluated by determining $u(\mathbf{y}_k)$ for all k. However, it is well known that such an approximation is very computationally costly in many practical problems. Instead we approximate the moments by means of a quadrature rule, i.e. the goal is to determine a finite number of nodes, denoted by the indexed set $X_N = \{\mathbf{x}_0, \ldots, \mathbf{x}_N\} \subset \Omega$, and weights, denoted by

$W_N = \{w_0, \dots, w_N\} \subset \mathbb{R}$ such that

$$\mathcal{I}^{(K)} u \approx \sum_{k=0}^{N} u(\mathbf{x}_k) w_k =: \mathcal{A}_N^{(K)} u.$$

The operator $\mathcal{A}_N^{(K)}$ is the quadrature rule operator using the nodal set X_N. We omit the number of samples K from the notation if it is clear from the context.

Three properties are relevant for quadrature rules: accuracy, positivity, and nesting. These properties have already been considered in Chapters 2 and 3. The extensions of these properties to the setting of this chapter are briefly discussed in Sections 4.2.1, 4.2.2, and 4.2.3. The terms *nodes* and *samples* are sometimes used interchangeably in a quadrature rule setting. This is not the case in this chapter: samples are elements from sample sets statistically describing a distribution (called Y_K), whereas nodes are the collocation points from a quadrature rule (called X_N).

4.2.1. Accuracy

Recall that in this thesis the accuracy of a quadrature rule is ensured by constructing it such that it integrates all polynomials $\varphi \in \Phi_D$ exactly, with $\Phi_D = \text{span}\{\varphi_0, \dots, \varphi_D\}$. Here, φ_j are basis polynomials with $\deg \varphi_j \leq \deg \varphi_k$ for $j \leq k$. The quadrature rule operator $\mathcal{A}_N^{(K)}$ is linear, hence if $D = N$ and K is given, the weights can be determined from the nodes by solving the following linear system:

$$\mathcal{A}_N^{(K)} \varphi_j = \mathcal{I}^{(K)} \varphi_j, \text{ for } j = 0, \dots, D. \tag{4.3}$$

In the univariate case, this linear system is non-singular if all nodes are distinct. This does not hold in general in the multivariate case or if $D \neq N$.

Similarly as to the previous chapters, the linear system (4.3) will be used often in this chapter to ensure the accuracy of the constructed quadrature rules. Recall that the matrix of this system is called the (multivariate) Vandermonde matrix, as introduced for univariate quadrature rule in (3.2) and denoted by $V_D(X_N)$. Its multivariate extension is straightforward: if a basis $\varphi_0, \dots, \varphi_D$ is given, the system of (4.3) can be written as

$$V_D(X_N)\mathbf{w} := \begin{pmatrix} \varphi_0(\mathbf{x}_0) & \cdots & \varphi_0(\mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ \varphi_D(\mathbf{x}_0) & \cdots & \varphi_D(\mathbf{x}_N) \end{pmatrix} \begin{pmatrix} w_0 \\ \vdots \\ w_N \end{pmatrix} = \begin{pmatrix} \mu_0^{(K)} \\ \vdots \\ \mu_D^{(K)} \end{pmatrix}. \tag{4.4}$$

Here, $\mu_j^{(K)}$ are known as the (multivariate) raw moments of the samples Y_K, i.e.

$$\mu_j^{(K)} := \frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(\mathbf{y}_k) = \mathcal{I}^{(K)} \varphi_j.$$

Notice that using samples to determine the moments is the key difference between the rules constructed in this chapter and the rules constructed in Chapter 3, where we assumed that exact values of the moments are known (recall μ_j as defined by (2.14) on page 17).

Throughout this chapter it is assumed that Φ_D is a polynomial space of minimal degree and that φ_k is a monomial for each k. Multivariate polynomials are sorted using the graded reverse lexicographic order. All methods discussed in this chapter can also be applied if the polynomials are sorted differently (i.e. a sparse or an orthonormal basis is considered) or if the basis under consideration is not polynomial at all (e.g. sinusoidal). The only imposed restriction is that φ_0 is the constant function.

The matrix $V_D(X_N)$ might become ill-conditioned if it is constructed using monomials even for small N. Since this matrix is used to construct quadrature rules in this chapter, this can limit the applicability of the methods discussed here. In this chapter, all quadrature rules have been constructed using (products of) Legendre polynomials, which resulted in a sufficiently well-conditioned matrix for moderately large N and D.

4.2.2. Positivity, stability, and convergence

Any constructed quadrature rule in this chapter has solely positive weights for two reasons: stability and convergence. Similarly as in Chapter 3, we call such a quadrature rule simply a *positive* quadrature rule. Both stability and convergence follow from the fact that the induced ∞-norm of $\mathcal{A}_N^{(K)}$ equals the sum of the absolute weights (which is, recalling (2.13), the condition number of $\mathcal{A}_N^{(K)}$ as $\mu_0^{(K)} = 1$), i.e.

$$\|\mathcal{A}_N^{(K)}\|_\infty := \sup_{\|u\|_\infty=1} |\mathcal{A}_N^{(K)} u| = \sum_{k=0}^{N} |w_k|, \text{ with } \|u\|_\infty := \max_{\mathbf{x}\in\Omega} |u(\mathbf{x})|.$$

This norm is minimal for quadrature rules with positive weights. In these cases, we have that for all K,

$$\|\mathcal{A}_N^{(K)}\|_\infty = \sum_{k=0}^{N} |w_k| = \sum_{k=0}^{N} w_k = 1 = \mathcal{I}^{(K)} 1.$$

Convergence can be demonstrated by applying the Lebesgue inequality [24], i.e. see (2.12). In this chapter, the following form is used:

$$|\mathcal{I}^{(K)} u - \mathcal{A}_N^{(K)} u| \leq \left(1 + \sum_{k=0}^{N} |w_k|\right) \inf_{\varphi\in\Phi_D} \|u-\varphi\|_\infty = 2 \inf_{\varphi\in\Phi_D} \|u-\varphi\|_\infty, \tag{4.5}$$

where it is used that all weights are non-negative.

The error of the quadrature rule $\mathcal{A}_N^{(K)} u$ with respect to $\mathcal{I}^{(K)} u$ does not depend on the accuracy of the moments $\mu_j^{(K)}$, i.e. on whether the number of samples is large enough to resolve $\mu_j^{(K)}$ accurately. This can be seen as follows. Assume the samples Y_K are drawn from a distribution $\rho\colon \Omega \to \mathbb{R}$ and let $\mathcal{I}$ be the integral from (4.2) weighted with this distribution. Even though $|\mathcal{I}\varphi_j - \mathcal{I}^{(K)}\varphi_j|$ can become large for increasing j, the error of the quadrature rule is not necessarily large:

$$|\mathcal{I} u - \mathcal{A}_N^{(K)} u| \leq \underbrace{|\mathcal{I} u - \mathcal{I}^{(K)} u|}_{\text{Sampling error}} + \underbrace{|\mathcal{I}^{(K)} u - \mathcal{A}_N^{(K)} u|}_{\text{Quadrature error}}. \tag{4.6}$$

The error depends on two components. The sampling error describes whether the number of samples is large enough to approximate the integral of u (which is independent

of φ_j), whereas the quadrature error describes whether the quadrature rule is accurate (which depends on φ_j, but not through the samples; see (4.5)). The quadrature error is conceptually different from the sampling error and often decreases much faster in N than the sampling error does in K. As we assume an arbitrarily sized sequence of samples is readily available to make the sampling error sufficiently small, this chapter will focus on the quadrature error.

4.2.3. Nesting

Nesting means that $X_{N_1} \subset X_{N_2}$ for some $N_1 < N_2$, i.e. the nodes of a smaller quadrature rule are contained in a larger quadrature rule. This allows for the reuse of model evaluations if the quadrature rule is refined by considering more nodes. We will call such a quadrature rule, with a little abuse of nomenclature, a *nested* quadrature rule (because strictly speaking it is a nested *sequence* of quadrature rules).

A nested quadrature rule has the favorable property that it can be used to provide an error estimate of the approximated integral. If the quadrature rule has positive weights and converges to the true integral, i.e. $\mathcal{A}_N u \to \mathcal{I} u$ for $N \to \infty$, then

$$\big||\mathcal{A}_{N_1} u - \mathcal{I} u| - |\mathcal{A}_{N_2} u - \mathcal{I} u|\big| \leq |\mathcal{A}_{N_1} u - \mathcal{A}_{N_2} u| \leq \big||\mathcal{A}_{N_1} u - \mathcal{I} u| + |\mathcal{A}_{N_2} u - \mathcal{I} u|\big|. \tag{4.7}$$

Hence the quantity $|\mathcal{A}_{N_1} u - \mathcal{A}_{N_2} u|$ can be used to estimate the accuracy of $A_{N_2} u$. If $X_{N_1} \subset X_{N_2}$, this error estimate is computable without any additional model evaluations.

4

4.3. The implicit quadrature rule

The implicit quadrature rule is a quadrature rule that is constructed using an arbitrarily sized sequence of samples. The crucial equation in the method is (4.3), which can be written as

$$\sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k = \mu_j^{(K)}, \text{ with } \mu_j^{(K)} = \frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(\mathbf{y}_k), \text{ for } j = 0, \dots, D. \tag{4.8}$$

Given a sequence of basis functions $\varphi_0, \dots, \varphi_D$, the left-hand side of this equation only depends on the quadrature nodes X_N and weights W_N, whereas the right-hand side of the equation only depends on the samples Y_K. The goal is to determine, based on the $K+1$ samples in the set Y_K, a subset of $N+1$ samples that form the nodes X_N of a quadrature rule in such a way that (4.8) is satisfied and such that the corresponding weights are positive. The existence of such a subset is motivated by the Tchakaloff bound [42], which states that there exists a quadrature rule with positive weights with $N = D$ if Φ_D encompasses polynomials (as being done in this chapter).

In principle, this problem can be solved using the removal step as explained in Section 3.3. Even though it has only been introduced in a univariate setting, it can be straightforwardly applied to remove columns from $V_D(Y_K)$ by using the definition from (4.4). Equivalently nodes can be removed from Y_K. Repeatedly removing nodes yields a quadrature rule that satisfies (4.8) with $N = D$. However, accurately computing a null vector of $V_D(Y_K)$ is often computationally intractable, considering that this is a $(K+1) \times (D+1)$-matrix and K is large.

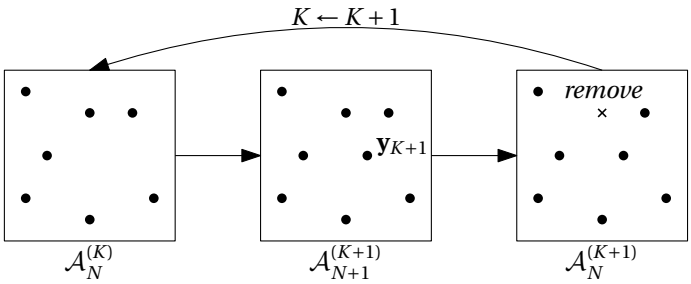


Figure 4.2: The implicit quadrature rule proposed in this chapter. Given a quadrature rule that integrates K samples ($\mathcal{A}_N^{(K)}$), a node $\mathbf{y}_{K+1}$ is added such that a rule is obtained of one more node ($\mathcal{A}_{N+1}^{(K+1)}$). Finally, one or more nodes are removed to obtain a quadrature rule of fewer nodes ($\mathcal{A}_N^{(K+1)}$), though the accuracy of the rule does not deteriorate.

To alleviate the high computational cost, an iterative algorithm is used to determine the quadrature rule: starting from an initial quadrature rule, the nodes and weights are changed iteratively while new samples $\mathbf{y}_k$ are added. Redundant nodes are removed while ensuring that the accuracy of the quadrature rule does not deteriorate. These two steps essentially encompass the addition and the removal of a node from the quadrature rule, in line with the framework set out in Chapter 3. This iterative step, which is the key idea of the proposed algorithm, is sketched in Figure 4.2. By repeatedly applying this step, a quadrature rule that validates (4.8) is obtained, without having to compute null vectors of a large matrix.

Our algorithm is explained in the next two sections. First, in Section 4.3.1 we propose a method for a slightly simpler problem: we fix D (or Φ_D) and determine at which nodes the model should be evaluated to integrate the sample moments while preserving positivity of weights. Second, in Section 4.3.2 this method is extended to create sequences of *nested* quadrature rules with increasing D, increasing K, or both. In other words, given N model evaluations, we determine a subset of the samples such that (4.8) is satisfied *and* the provided N model evaluations are reused.

4.3.1. Non-nested implicit rule

The goal is to construct a positive quadrature rule that integrates all $\varphi \in \Phi_D$ exactly, where D is provided a priori. The quadrature rule will consist of (at most) $D+1$ nodes in this case. Without loss of generality, it is assumed that $D < K$, i.e. the number of available samples is at least as large as the dimension of Φ_D.

The initial step is to consider Y_D and to construct the following quadrature rule for $N = D$:

$$X_N^{(D)} = Y_D = \{\mathbf{y}_0, \ldots, \mathbf{y}_D\},$$
$$W_N^{(D)} = \left\{\frac{1}{D+1}, \ldots, \frac{1}{D+1}\right\}.$$

The upper index describes the set of samples used for the construction, in this case Y_D, and the lower index describes the number of nodes of the quadrature rule (i.e. $\mathbf{x}_0, \ldots, \mathbf{x}_N$). This initial rule simply approximates the moments by means of Monte Carlo and it is obvious that (4.8) holds for $K = D$.

The iterative procedure works as follows. Assume $X_N^{(K)}$, $W_N^{(K)}$ form the positive quadrature rule integrating all $\varphi \in \Phi_D$ exactly as if Y_K was used (i.e. (4.8) holds). This quadrature rule has the property that $\mathcal{A}_N^{(K)} \varphi_j = \mu_j^{(K)}$ for $j = 0, \ldots, D$. The goal is to construct a quadrature rule that also has this property, but with the moments $\mu_j^{(K+1)}$ as the right-hand side. To this end, let $\mathbf{y}_{K+1}$ be the next sample and straightforwardly determine $X_{N+1}^{(K+1)}$ and $W_{N+1}^{(K+1)}$ as follows:

$$\begin{aligned} X_{N+1}^{(K+1)} &= X_N^{(K)} \cup \{\mathbf{y}_{K+1}\}, \\ W_{N+1}^{(K+1)} &= \left(\left(\frac{K+1}{K+2} \right) \cdot W_N^{(K)} \right) \cup \left\{ \frac{1}{K+2} \right\}, \end{aligned} \tag{4.9}$$

i.e. $\mathbf{y}_{K+1}$ is "added" to $X_N^{(K)}$ (hence $\mathbf{x}_{N+1} = \mathbf{y}_{K+1}$) and the weights are changed such that the quadrature rule again integrates the sample moments. The latter can be seen as follows:

$$\begin{aligned} \sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) \frac{K+1}{K+2} w_k + \frac{1}{K+2} \varphi_j(\mathbf{x}_{N+1}) &= \frac{K+1}{K+2} \sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k + \frac{1}{K+2} \varphi_j(\mathbf{x}_{N+1}) \\ &= \frac{K+1}{K+2} \left(\frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(\mathbf{y}_k) \right) + \frac{1}{K+2} \varphi_j(\mathbf{y}_{K+1}) \\ &= \frac{1}{K+2} \sum_{k=0}^{K+1} \varphi_j(\mathbf{y}_k) = \mu_j^{(K+1)}. \end{aligned}$$

Here, w_k are the weights from the *original* quadrature rule, i.e. $w_k \in W_N^{(K)}$. We will use v_k to denote the weights from the updated quadrature rule, i.e. $v_k \in W_{N+1}^{(K+1)}$.

If $W_N^{(K)}$ consists of positive weights, then so does $W_{N+1}^{(K+1)}$. The problem with this simple update is that, compared to the original nodal set, the quadrature rule now requires an additional node to integrate all $\varphi \in \Phi_D$ exactly (i.e. (4.8) holds), resulting to a total of $N+2$ nodes.

In order to construct a quadrature rule that requires only $N+1$ nodes (while preserving positive weights and integrating $\mu_j^{(K+1)}$ exactly), one node will be removed from the extended rule $X_{N+1}^{(K+1)}$, following the procedure outlined in Chapter 3. The procedure is briefly repeated, since the setting in this chapter is significantly more general than the setting in where the procedure was introduced. It still follows the proof of Carathéodory's theorem, as outlined in Theorem 3.1 on page 33.

The Vandermonde matrix of the extended quadrature rule, i.e. $V_D(X_{N+1}^{(K+1)})$, is as follows:

$$V_D(X_{N+1}^{(K+1)}) = \begin{pmatrix} \varphi_0(\mathbf{x}_0) & \ldots & \varphi_0(\mathbf{x}_N) & \varphi_0(\mathbf{x}_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(\mathbf{x}_0) & \ldots & \varphi_D(\mathbf{x}_N) & \varphi_D(\mathbf{x}_{N+1}) \end{pmatrix}.$$

This is a $(D+1)\times(N+2)$-matrix (with $N=D$), so at least one non-trivial null vector $\mathbf{c}=(c_0,\dots,c_{N+1})^{\mathrm{T}}$ of this matrix exists. Moreover, any multiple of this null vector is also a null vector. By combining this with (4.8), the following linear system is obtained for arbitrary $\alpha\in\mathbb{R}$:

$$\begin{pmatrix} \varphi_0(\mathbf{x}_0) & \dots & \varphi_0(\mathbf{x}_N) & \varphi_0(\mathbf{x}_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(\mathbf{x}_0) & \dots & \varphi_D(\mathbf{x}_N) & \varphi_D(\mathbf{x}_{N+1}) \end{pmatrix} \begin{pmatrix} v_0-\alpha c_0 \\ \vdots \\ v_N-\alpha c_N \\ v_{N+1}-\alpha c_{N+1} \end{pmatrix} = \begin{pmatrix} \mu_0^{(K+1)} \\ \vdots \\ \mu_D^{(K+1)} \end{pmatrix}.$$

This equation can be interpreted as a quadrature rule depending on the free parameter α with nodes $X_{N+1}^{(K+1)}$ and weights $\{v_k-\alpha c_k \mid k=0,\dots,N+1\}$. The parameter α can be used to remove one node from the quadrature rule, as nodes with weight equal to zero can be removed from the quadrature rule without deteriorating it. There are two options, $\alpha=\alpha_1$ or $\alpha=\alpha_2$:

$$\begin{aligned} \alpha_1 &= \min_k\left(\frac{v_k}{c_k} \mid c_k>0\right) =: \frac{v_{k_1}}{c_{k_1}}, \\ \alpha_2 &= \max_k\left(\frac{v_k}{c_k} \mid c_k<0\right) =: \frac{v_{k_2}}{c_{k_2}}. \end{aligned}$$

The sets $\{v_k-\alpha_1 c_k\}$ and $\{v_k-\alpha_2 c_k\}$ consist of non-negative weights and (at least) one weight equal to zero. Both α_1 and α_2 are well defined, because $\mathbf{c}$ has both positive and negative elements. The latter follows from the fact that φ_0 is assumed to be a constant and that $\mathbf{c}$ is not equal to the zero vector, i.e.

$$0=\sum_{k=0}^{N+1}\varphi_0(\mathbf{x}_k)c_k=\varphi_0\sum_{k=0}^{N+1}c_k.$$

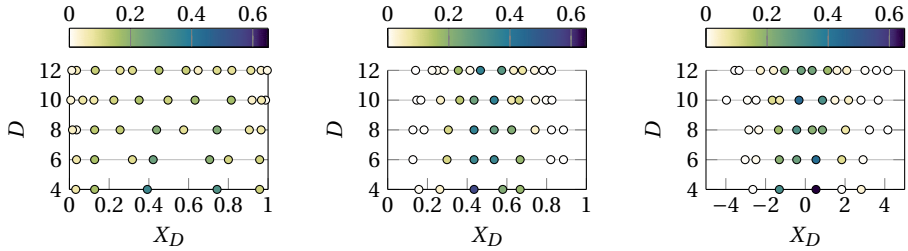
The desired quadrature rule that integrates all $\varphi\in\Phi_D$ exactly and consists of $N=D$ nodes can be constructed by choosing either $i=1$ or $i=2$ and determining the nodes and weights as follows:

$$\begin{aligned} X_N^{(K+1)} &= X_{N+1}^{(K+1)}\setminus\{\mathbf{x}_{k_i}\}, \\ W_N^{(K+1)} &= \{v_k-\alpha_i c_k \mid k=0,\dots,k_i-1,k_i+1,\dots,N+1\}. \end{aligned}$$

This rule has $N+1$ nodes and integrates the moments $\mu_j^{(K+1)}$ for $j=0,\dots,D$ exactly. Note that, to include the case of two weights becoming zero simultaneously (which is the case for symmetric quadrature rules [15]), these sets can be implicitly defined as follows:

$$\begin{aligned} X_Q^{(K+1)} &= \left\{\mathbf{x}_k \mid \mathbf{x}_k\in X_{N+1}^{(K+1)} \text{ and } v_k>\alpha_i c_k\right\}, \\ W_Q^{(K+1)} &= \left\{v_k-\alpha_i c_k \mid v_k\in W_{N+1}^{(K+1)} \text{ and } v_k>\alpha_i c_k\right\}, \end{aligned}$$

with $Q\le N\le D$. Without loss of generality, we assume $Q=N$ throughout this chapter.



(a) $U[0,1]$ **(b)** $\beta(10,10)$ **(c)** $\mathcal{N}(0,1)$

Figure 4.3: Examples of implicit quadrature rules for various degrees, using the same 10^5 samples for each degree. The colors indicate the weights of the nodes.

The correctness of this method follows from the fact that the first $D+1$ sample moments of the first K samples are integrated exactly using the constructed quadrature rule after iteration K. All quadrature rules constructed this way are interpolatory, since they consist of $N+1$ nodes and integrate all functions in the $(N+1)$-dimensional space Φ_N exactly. Therefore by construction the following theorem is proved.

Theorem 4.1. *Let $\mathcal{A}_N^{(K)}$ be a positive quadrature rule operator such that*

$$\mathcal{A}_N^{(K)}\varphi_j = \mu_j^{(K)}, \text{ for } j = 0,\dots,D,$$

with $N = D$. Then after applying the procedure above, a positive quadrature rule operator $\mathcal{A}_N^{(K+1)}$ is obtained such that

$$\mathcal{A}_N^{(K+1)}\varphi_j = \mu_j^{(K+1)}, \text{ for } j = 0,\dots,D.$$

For different sample sets, even when drawn from the same distribution, the procedure constructs different quadrature rules. If desired, this non-deterministic nature of the quadrature rule can be eradicated by using deterministic samplers, such as quasi-Monte Carlo sequences [31], as briefly discussed in Section 2.1.1. These are not used in the quadrature rules constructed in this chapter, as these sequences are generally not straightforward to construct for distributions with a non-invertible cumulative distribution function. Another aspect of the algorithm that can create variation in the resulting quadrature rules is the choice of the parameter α. It is possible to incorporate knowledge about the integrand in the choice for α at each iteration. In this chapter the smallest value is used and it is assumed that we do not have a priori knowledge about the integrand.

The steps of the method are outlined in Algorithm 4.1 and examples of implicit quadrature rules obtained using sample sets drawn from well-known distributions are depicted in Figures 4.3 and 4.4. In Figure 4.3, the nodes and weights are shown for various polynomial degrees D, based on $K_{\max} = 10^5$ samples drawn from several common univariate distributions. For the distributions with compact support, the nodes cluster at the boundaries of the domain. However, the nodes exhibit an irregular

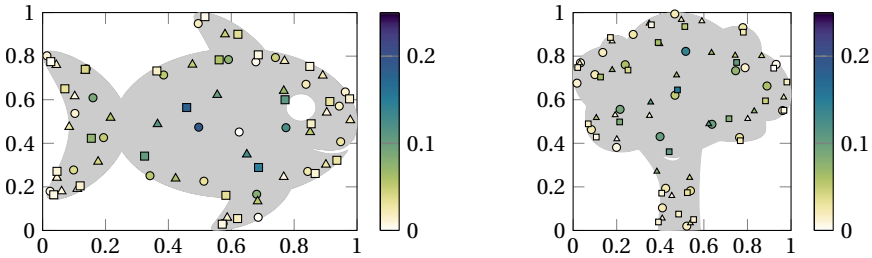


Figure 4.4: Three implicit quadrature rules of 25 nodes (using different symbols) determined using the bivariate uniform distribution restricted to the gray area, using three different permutations of a set of 10^5 samples. The colors indicate the weights of the nodes.

pattern upon increasing the degree, and determining a quadrature rule with a higher degree does not result in a nested rule (this will be addressed in Section 4.3.2). In the second example, nodal sets are generated in two dimensions on two different irregular domains, see Figure 4.4. This shows a major strength of the proposed implicit quadrature rule: it can be applied to arbitrary sample sets, including domains that are not simply connected, and positive weights are still guaranteed. Depending on the ordering of the samples in the set, different nodes and weights are obtained, indicating that the quadrature rules for these sets are not unique. It is generally not possible to obtain exactly the same quadrature rule for two permutation of the sample set, since choosing either α_1 or α_2 can be exploited to preserve only a single node in the rule. Theoretically this can be resolved by removing multiple nodes from the rule, as will be done in the next section, though it is often unfeasible to do so.

4.3.2. Nested implicit rule

The approach of the previous section can be used to construct a quadrature rule given the number of basis vectors D and a fixed number of samples K. For varying D these quadrature rules are, however, not nested. In this section the algorithm is extended such that the constructed quadrature rules contain nodes that can be provided beforehand. By providing the nodes of an existing quadrature rule, a sequence of nested quadrature rules can be constructed.

The problem setting is as follows. Let X_N be an indexed set of quadrature rule nodes and assume D is specified, with $D \geq N$. The goal is to add M nodes to X_N in order to obtain a positive quadrature rule with nodes X_{N+M} (so $X_N \subset X_{N+M}$) that integrates all $\varphi \in \Phi_D$ exactly. Note that in general all weights will differ, i.e. $W_N \not\subset W_{N+M}$. We desire to add a small number of nodes, thus M to be small, but it is straightforward to observe that M is bounded as follows:

$$D \leq N + M \leq N + D + 1. \tag{4.10}$$

The first bound ($D \leq N + M$) describes that the number of basis vectors a quadrature rule constructed with our algorithms integrates exactly is not larger than its number of

Algorithm 4.1: The implicit quadrature rule

Input: Samples $\{\mathbf{y}_0, \dots, \mathbf{y}_{K_{\max}}\}$, basis polynomials $\{\varphi_0, \dots, \varphi_D\}$
Output: Positive quadrature rule $X_N = \{\mathbf{x}_0, \dots, \mathbf{x}_N\}$, $W_N = \{w_0, \dots, w_N\}$ with $N = D$

Initialize $X_N^{(D)} = \{\mathbf{y}_0, \dots, \mathbf{y}_D\}$
Initialize $W_N^{(D)} = \{1/(D+1), \dots, 1/(D+1)\}$
for $K = D, \dots, K_{\max} - 1$ **do**
  Add node:
  $X_{N+1}^{(K+1)} \leftarrow X_N^{(K)} \cup \{\mathbf{y}_{K+1}\}$
  $W_{N+1}^{(K+1)} \leftarrow (K+1)/(K+2) W_N^{(K)} \cup \{1/(K+2)\}$

  Update weights:
  Construct $V_D(X_{N+1}^{(K+1)})$
  Determine (non-trivial) $\mathbf{c}$ such that $V_D(X_{N+1}^{(K+1)})\mathbf{c} = \mathbf{0}$
  $\alpha_1 \leftarrow \min_k(v_k/c_k \mid c_k > 0)$, with $v_k \in W_{N+1}^{(K+1)}$
  $\alpha_2 \leftarrow \max_k(v_k/c_k \mid c_k < 0)$, with $v_k \in W_{N+1}^{(K+1)}$

  Choose:
  Either $\alpha \leftarrow \alpha_1$ or $\alpha \leftarrow \alpha_2$

  Remove node:
  $X_N^{(K+1)} \leftarrow \left\{\mathbf{x}_k \mid \mathbf{x}_k \in X_{N+1}^{(K+1)} \text{ and } w_k > \alpha c_k\right\}$
  $W_N^{(K+1)} \leftarrow \left\{w_k - \alpha c_k \mid w_k \in W_{N+1}^{(K+1)} \text{ and } w_k > \alpha c_k\right\}$
end for
Return $X_N^{(K_{\max})}$, $W_N^{(K_{\max})}$

nodes. The second bound ($N + M \leq N + D + 1$) describes that it is possible to simply add a quadrature rule with $D+1$ nodes to the existing quadrature rule by setting all existing weights to 0. This is often not desired in applications but provides a theoretical bound on the number of nodes obtained using our algorithms.

Algorithm 4.1 can be straightforwardly extended to incorporate nodes that are provided beforehand. The algorithm proceeds as usual, with the difference that nodes can be removed only if they were *added* during the algorithm, but not if *provided* in advance. This approach yields a sequence of nested quadrature rules, but is not optimal because it results in a quadrature rule with (possibly many) more nodes than necessary. In such a case the null space of the Vandermonde matrix $V_D(X_N)$ is multidimensional. Hence there might exist multiple nodes that can be removed together, even though removing the nodes individually yields a quadrature rule with negative weights. For example, removing two nodes from the rule yields a positive rule, but removing only one of the two yields a negative rule.

In this section the focus is therefore on the removal step of Algorithm 4.1, which is extended to incorporate the removal of multiple nodes. By combining such an algorithm

with Algorithm 4.1 the nested implicit quadrature rule is obtained.

Sequentially removing multiple nodes that result in a positive quadrature rule can result in intermediate quadrature rules with negative weights. Therefore the first step is to extend the removal procedure outlined in Section 4.3.1 such that it supports negative weights. The main algorithm that generalizes the approach of the basic implicit rule follows readily. With this algorithm, a procedure is obtained to determine a nested implicit quadrature rule.

Negative weight removal

The procedure from the previous section determines α_1 and α_2 that can be used for the removal of a node. However, the equations for α_1 and α_2 were derived assuming positive weights. In this section, similar equations will be derived without assuming positive weights.

In this chapter, the derived equations are used to facilitate the removal of multiple nodes such that a quadrature rule of minimal size is obtained. Notice that these results can straightforwardly be embedded in the framework discussed in Chapter 3 or, more specifically, in the removal procedure outlined in Section 3.3.

Let X_N, W_N be a quadrature rule with (possibly) negative weights. The goal is to remove one node to obtain X_{N-1} and W_{N-1} such that the resulting quadrature rule has positive weights and $\mathcal{A}_{N-1}\varphi_j = \mathcal{A}_N\varphi_j$ for $j = 0,\dots,N-1$. As introduced before, let $V_{N-1}(X_N)$ be the respective $N\times(N+1)$ Vandermonde matrix and let $\mathbf{c}\in\mathbb{R}^{N+1}$ be a non-trivial null vector of that matrix. The goal is to have only positive weights, hence with the same reasoning as before, we obtain the following bound:

$$w_k - \alpha c_k \geq 0, \text{ for all } k \text{ and a certain } \alpha.$$

This translates into two cases:

$$\alpha \begin{cases} \geq w_k/c_k & \text{for all } k \text{ with } c_k < 0, \\ \leq w_k/c_k & \text{for all } k \text{ with } c_k > 0. \end{cases}$$

Hence the following bounds should hold for any such α:

$$\begin{aligned} &\alpha_{\min} \leq \alpha \leq \alpha_{\max}, \text{ with} \\ &\alpha_{\min} = \max_k \left(\frac{w_k}{c_k} \mid c_k < 0 \right) =: \frac{w_{k_{\min}}}{c_{k_{\min}}}, \\ &\alpha_{\max} = \min_k \left(\frac{w_k}{c_k} \mid c_k > 0 \right) =: \frac{w_{k_{\max}}}{c_{k_{\max}}}. \end{aligned} \tag{4.11}$$

Such α does not necessarily exist, but if it does, either $\alpha = \alpha_{\min}$ or $\alpha = \alpha_{\max}$ can be used to remove either the node $\mathbf{x}_{k_{\min}}$ or $\mathbf{x}_{k_{\max}}$ respectively from the rule (as their weight becomes 0). The case with only positive weights (which was considered in Section 4.3.1) fits naturally in this, with $\alpha_1 = \alpha_{\max}$ and $\alpha_2 = \alpha_{\min}$. If all weights are positive, it is evident that $\alpha_{\min} < 0 < \alpha_{\max}$.

Even a stronger, less trivial result holds: if $\alpha_{\max} < \alpha_{\min}$, then *no* node exists that results in a positive quadrature rule after removal and if $\alpha_{\min} < \alpha_{\max}$, there exist *exactly*

two nodes such that removing one of the two results in a quadrature rule with positive weights. In other words, determining $\alpha_{\min}$ and $\alpha_{\max}$ as above yields all possible nodes that can be removed resulting into a positive quadrature rule. The details are discussed in the proof of the following lemma, which also proves Theorem 3.2 (see page 34).

Lemma 4.2. *Let X_N, W_N be a quadrature rule integrating all $\varphi \in \Phi_N$ exactly. The following statements are equivalent:*

1. $\alpha_{\min} \leq \alpha_{\max}$.
2. *There exists an $\mathbf{x}_{k_0} \in X_N$ such that the quadrature rule with nodes $X_N \setminus \{\mathbf{x}_{k_0}\}$ that integrates all $\varphi \in \Phi_{N-1}$ exactly has non-negative weights.*
3. *Let any $\mathbf{x}_{k_0} \in X_N$ be given such that the quadrature rule with nodes $X_N \setminus \{\mathbf{x}_{k_0}\}$ that integrates all $\varphi \in \Phi_{N-1}$ exactly has non-negative weights. Then the weights of this rule, say W_{N-1}, are formed by*

$$W_{N-1} = \{w_k - \alpha c_k \mid k \neq k_0\},$$

where c_k are the elements of a null vector of $V_{N-1}(X_N)$ and either $\alpha = \alpha_{\min}$ or $\alpha = \alpha_{\max}$.

Proof. The proof consists of three parts: 1 → 2 → 3 → 1.

(1 → 2) The proof of 1 → 2 follows immediately from the removal step outlined above, i.e. see (4.11).

(2 → 3) Suppose 2 holds and let $\mathbf{x}_{k_0}$ be given. Without loss of generality assume $k_0 = N$. Let W_{N-1} be the weights of the quadrature rule nodes $X_{N-1} = X_N \setminus \{\mathbf{x}_N\}$ and let $w_k^{(N)} \in W_N$ and $w_k^{(N-1)} \in W_{N-1}$. It holds that the nodes X_N and weights W_N form a quadrature rule that integrates all $\varphi \in \Phi_N$ exactly and that the nodes X_{N-1} and weights W_{N-1} form a quadrature rule that integrates all $\varphi \in \Phi_{N-1}$ exactly. Therefore for $j = 0, \ldots, N-1$ the following holds:

$$\sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k^{(N)} = \sum_{k=0}^{N-1} \varphi_j(\mathbf{x}_k) w_k^{(N-1)},$$

so for these j it follows that

$$\sum_{k=0}^{N-1} \varphi_j(\mathbf{x}_k)(w_k^{(N)} - w_k^{(N-1)}) + \varphi_j(\mathbf{x}_N) w_N^{(N)} = 0.$$

Hence the vector $\mathbf{c} \in \mathbb{R}^{N+1}$ with elements $c_k = w_k^{(N)} - w_k^{(N-1)}$ (and $c_N = w_N^{(N)}$) is a null vector of $V_{N-1}(X_N)$. Then it follows that $\alpha = 1$. Without loss of generality, assume that $c_{k_0} \neq 0$.

The remainder of this part consists of demonstrating that either $\alpha = \alpha_{\min}$ or $\alpha = \alpha_{\max}$. Assume $c_{k_0} > 0$ (with $k_0 = N$). It holds that $w_{k_0}^{(N)} = c_{k_0}$ and $w_k^{(N)} \geq c_k$ for all other k. For all k with $c_k > 0$ (including k_0), we therefore obtain

$$\frac{w_k^{(N)}}{c_k} \geq 1.$$

Equality is attained at $k = k_0$, hence $1 = \min(w_k^{(N)}/c_k \mid c_k > 0) = \alpha_{\max}$. In a similar way it can be demonstrated that if $c_{k_0} < 0$, we have $1 = \max(w_k^{(N)}/c_k \mid c_k < 0) = \alpha_{\min}$, concluding this part of the proof.

($3 \to 1$) Suppose 3 holds and let the weights be given as in the lemma. Let $\alpha_{\min}$, $\alpha_{\max}$, $k_{\min}$ and $k_{\max}$ be given. By definition of $\alpha_{\max}$, it holds that $c_{k_{\max}} > 0$ (see (4.11)). So if $\alpha \geq \alpha_{\max}$, then $w_{k_{\max}} \leq \alpha c_{k_{\max}}$. Hence to have positive weights, we must have $\alpha \leq \alpha_{\max}$.

Similarly we have that $c_{k_{\min}} < 0$ and therefore if $\alpha \leq \alpha_{\min}$, it holds that $w_{k_{\min}} \leq \alpha c_{k_{\min}}$. So to have positive weights, we must have $\alpha \geq \alpha_{\min}$.

If there exists an α such that $\alpha_{\min} \leq \alpha$ and $\alpha \leq \alpha_{\max}$, it must hold that $\alpha_{\min} \leq \alpha_{\max}$. ■

The lemma demonstrates that $\alpha_{\min}$ and $\alpha_{\max}$ from (4.11) can be used to determine whether there exists a node that yields a positive quadrature rule after removal (i.e. if $\alpha_{\min} \leq \alpha_{\max}$) and if such a node exists, either $\alpha_{\min}$ or $\alpha_{\max}$ can be used to determine it (by determining k_0 as in the proof). If $\alpha_{\max} > \alpha_{\min}$, no such node exists, which is not an issue, since the algorithm to construct quadrature rules discussed in this chapter does by construction not end up in this case.

Removal of multiple nodes

Let X_N and W_N form a positive quadrature rule. In this section, the goal is to determine all subsets of M nodes that have one specific property in common: removing those M nodes results in a positive quadrature rule of $N+1-M$ nodes that exactly integrates all $\varphi \in \Phi_{N-M}$. We call a subset with this property an M-removal. Hence in the previous section a procedure has been presented to determine all 1-removals.

Lemma 4.2 is the main ingredient for deriving all M-removals. The idea boils down to the following. Let an M-removal be given, say $(\mathbf{q}_1, \ldots, \mathbf{q}_M) \subset X_N$. If the first $M-1$ nodes from this M-removal are removed, the M-th node $\mathbf{q}_M$ can be determined straightforwardly using $\alpha_{\min}$ or $\alpha_{\max}$ from (4.11). There are two possible values of α (namely either $\alpha_{\min}$ or $\alpha_{\max}$), hence there exists a second node, say $\widehat{\mathbf{q}}_M$, such that $(\mathbf{q}_1, \ldots, \mathbf{q}_{M-1}, \widehat{\mathbf{q}}_M)$ is also an M-removal. The order in which the nodes are removed is irrelevant, so each node $\mathbf{q}_k$ can be replaced in this way by a different node $\widehat{\mathbf{q}}_k$ resulting in a valid M-removal, i.e. a set of M nodes that can be removed while preserving positive weights and obtaining a quadrature rule that exactly integrates all $\varphi \in \Phi_{N-M}$.

We denote the procedure of obtaining a different M-removal from an existing one by the operator $F\colon [X_N]^M \to [X_N]^M$, where $[X_N]^M$ denotes the set of all M-subsets of X_N. If $(\mathbf{q}_1, \ldots, \mathbf{q}_M)$ is an M-removal, applying F yields the M-removal $(\mathbf{q}_1, \ldots, \mathbf{q}_{M-1}, \widehat{\mathbf{q}}_M)$. Such an operator is well defined, since Lemma 4.2 prescribes that there exist exactly two M-removals whose first $M-1$ elements equal $\mathbf{q}_1, \ldots, \mathbf{q}_{M-1}$. Notice that the operator F, which depends on the nodes and weights of the quadrature rule, can be computed by determining $\alpha_{\min}$ and $\alpha_{\max}$ from Lemma 4.2 after removal of $\mathbf{q}_1, \ldots, \mathbf{q}_{M-1}$.

By permuting the M-removal before applying F, one M-removal yields (up to a permutation at most) M other M-removals. These M-removals can be considered in a similar fashion and recursively more M-removals can be determined. This procedure yields all M-removals, which is demonstrated in the following lemma.

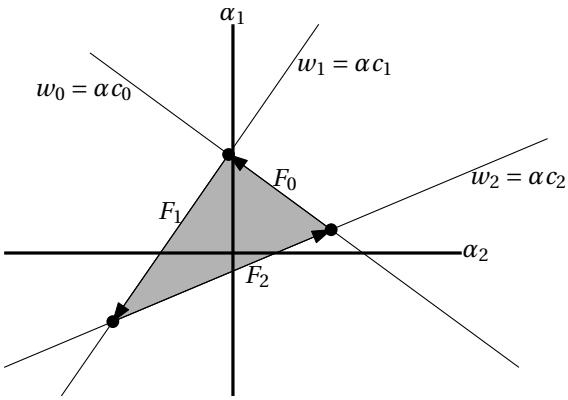


Figure 4.5: Graphical sketch of the simplex describing the removal of two nodes from a quadrature rule of three nodes. The gray area describes the simplex where all values of (α_1, α_2) yield positive weights. The operator F_k (see proof of Lemma 4.3) can be used to traverse the boundary of the simplex.

Lemma 4.3. *Let $(\mathbf{q}_1, \dots, \mathbf{q}_M)$ and $(\mathbf{s}_1, \dots, \mathbf{s}_M)$ be any two different M-removals of the positive quadrature rule X_N, W_N. Let the operator $F\colon [X_N]^M \to [X_N]^M$, as described in the text, be such that*

$$F(\mathbf{q}_1, \dots, \mathbf{q}_{M-1}, \mathbf{q}_M) = (\mathbf{q}_1, \dots, \mathbf{q}_{M-1}, \widehat{\mathbf{q}}_M),$$

for a given M-removal $(\mathbf{q}_1, \dots, \mathbf{q}_M)$, i.e. it replaces $\mathbf{q}_M$ by $\widehat{\mathbf{q}}_M$ such that $(\mathbf{q}_1, \dots, \mathbf{q}_{M-1}, \widehat{\mathbf{q}}_M)$ is an M-removal. Then there exists a finite number of permutations $\sigma_1, \dots, \sigma_n$ such that

$$(\sigma_1 \circ F \circ \sigma_2 \circ F \circ \cdots \circ F \circ \sigma_{n-1} \circ F \circ \sigma_n)(\mathbf{q}_1, \dots, \mathbf{q}_M) = (\mathbf{s}_1, \dots, \mathbf{s}_M).$$

Proof. Let $F_k\colon [X_N]^M \to [X_N]^M$ be the operator that firstly permutes $\mathbf{q}_k$ to the end of the M-removal and secondly applies F, i.e.

$$F_k(\mathbf{q}_1, \dots, \mathbf{q}_M) = F(\mathbf{q}_1, \dots, \mathbf{q}_{k-1}, \mathbf{q}_{k+1}, \dots, \mathbf{q}_M, \mathbf{q}_k) = (\mathbf{q}_1, \dots, \mathbf{q}_{k-1}, \mathbf{q}_{k+1}, \dots, \mathbf{q}_M, \widehat{\mathbf{q}}_k).$$

Notice that $F_k = F \circ \pi_k$, where π_k denotes the permutation that appends $\mathbf{q}_k$. Hence if there exist $k_1, \dots, k_n$ such that $F_{k_1} \circ \cdots \circ F_{k_n}(\mathbf{q}_1, \dots, \mathbf{q}_M)$ equals $(\mathbf{s}_1, \dots, \mathbf{s}_M)$ up to a permutation, the proof is done.

Consider $W_N = \{w_0, \dots, w_N\}$ and let $V_{N-M}(X_N)$ be the Vandermonde matrix with respect to this quadrature rule. This is an $(N-M+1) \times (N+1)$-matrix, so there exist M linearly independent null vectors $\mathbf{c}^1, \dots, \mathbf{c}^M \in \mathbb{R}^{N+1}$. We use the following notation for the vector $\mathbf{c}^k$:

$$\mathbf{c}^k = (c_0^k, \dots, c_N^k)^{\mathrm{T}}.$$

Let $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_M)$ be an M-tuple and consider the following set:

$$S = \left\{ \boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_M) \;\middle|\; \text{for all } k = 0, \dots, N \text{ with } w_k - \sum_{j=1}^{M} \alpha_j c_k^j \geq 0 \right\}.$$

The set S is a closed simplex, as it is formed by a finite number of linear inequalities. Moreover it is non-empty, as $(0,\dots,0) \in S$ (this follows from the fact that $w_k \geq 0$ for all k).

The boundary of S, i.e. ∂S, is of special interest. If $(\alpha_1,\dots,\alpha_M) \in \partial S$, it holds that $w_k - \sum_{j=1}^{M} \alpha_j c_k^j = 0$ for at least one k. At vertices of the simplex the highest number of weights, namely M, is 0. The operator F_k can be used to traverse the vertices of the simplex. The simplex S for determining a 2-removal for a quadrature rule of three nodes is sketched in Figure 4.5.

Consider an M-removal $(\mathbf{q}_1,\dots,\mathbf{q}_M)$. There is exactly one M-tuple $\boldsymbol{\alpha} = (\alpha_1,\dots,\alpha_M)$ resulting into the removal of these nodes. These α's coincide with a vertex of simplex S. Applying F_k to $(\mathbf{q}_1,\dots,\mathbf{q}_M)$ yields a different M-tuple. These two M-tuples are connected through an edge of the simplex. Due to Lemma 4.2 all M-tuples that are connected to $\boldsymbol{\alpha}$ through an edge can be found. Therefore, for a given vertex, the operator F_k yields all connected vertices and can be used to traverse the boundary of the simplex. This concludes the proof. ■

The lemma is constructive: given a single M-removal, all M-removals can be found. By repetitively constructing a 1-removal, an initial M-removal can be obtained (which is a vertex of the simplex S discussed in the proof). Lemma 4.3 ensures that any other M-removal can be reached from this removal. An outline of this procedure can be found in Algorithm 4.2. The computational cost of calculating the null vectors can be alleviated by decomposing $V_{N-M}(X_{N-M})$ once (e.g. by using an LU or QR decomposition) and computing the null vectors of $V_{N-M}(X_{N-M+1})$ in the loop by reusing this decomposition.

The time complexity of this algorithm is $\mathcal{O}(Z \log Z + Z(N-M)^3)$, where Z is the number of M-removals. Here, the term $Z \log Z$ originates from storing all visited M-removals efficiently using a binary search tree (which results in Z lookups that scale with $\log Z$) and the term $Z(N-M)^3$ is obtained by factorizing $V_{N-M}(X_{N-M})$ and repeatedly computing the null vector of $V_{N-M}(X_{N-M+1})$ using this factorization. Algorithm 4.2 always terminates, as the number of subsets of M nodes is strictly bounded. Combining this with the proof of Lemma 4.3 proves the following theorem.

Theorem 4.4. *On termination, Algorithm 4.2 returns all M-removals of the positive quadrature rule X_N, W_N.*

Theoretically, Algorithm 4.2 can be used to determine *all* quadrature rules $\mathcal{A}_N^{(K)}$ with $\mathcal{A}_N^{(K)}\varphi = \mathcal{I}^{(K)}\varphi$ for all $\varphi \in \Phi_D$. All these rules are obtained by computing all M-removals with $M = K - N$ of the quadrature rule $X_K = Y_K$ with $w_k = 1/(K+1)$ for all $k = 0,\dots,K$. However, in practice this is intractable, as the number of M-removals grows rapidly in M and $K-N$ is typically a large quantity.

The nested implicit quadrature rule

In this section the key algorithm of this chapter (and maybe even of this thesis) is presented, namely the nested implicit quadrature rule for arbitrary sample sets. It is constructed by combining the algorithms from the previous sections. Given a quadrature rule, two different refinements (or a combination of both) are considered. First, the number of samples K can be increased to obtain a more accurate estimate of $\mu_j^{(K)}$. Second, D can be increased to obtain a more accurate quadrature rule.

Algorithm 4.2: Removing multiple nodes

Input: Positive quadrature rule X_N, W_N, integer M with $1 \le M < N+1$
Output: All M-removals of X_N, W_N

Construct $V_{N-M}(X_N)$
Determine M independent null vectors $\mathbf{c}^j$ $(j = 1,\dots,M)$ of $V_{N-M}(X_N)$
Construct an M-removal, say $Q \leftarrow (\mathbf{q}_1,\dots,\mathbf{q}_M) \subset X_N$ (by repeatedly using Lemma 4.2)
$I \leftarrow \{Q\}$, the set containing all *queued* removals
$R \leftarrow \emptyset$, the set containing all *processed* removals
while $I \neq \emptyset$ **do**
    Get the first removal from I, say $Q \leftarrow (\mathbf{q}_1,\dots,\mathbf{q}_M) \subset X_N$
    Remove Q from I, i.e. $I \leftarrow I \setminus \{Q\}$.
    for $i = 1,\dots,M$ **do**
        Construct X_{N-M+1} and W_{N-M+1} by removing $(\mathbf{q}_1,\dots,\mathbf{q}_{i-1},\mathbf{q}_{i+1},\dots,\mathbf{q}_M)$
        Determine $\mathbf{c}$ such that $V_{N-M}(X_{N-M+1})\mathbf{c} = \mathbf{0}$
        Determine $\alpha_{\min}$, $\alpha_{\max}$, $k_{\min}$, $k_{\max}$ from (4.11)
        if $\mathbf{q}_i = \mathbf{x}_{k_{\max}}$ **then**
            $\widehat{\mathbf{q}}_i \leftarrow \mathbf{x}_{k_{\min}}$
        else
            $\widehat{\mathbf{q}}_i \leftarrow \mathbf{x}_{k_{\max}}$
        end if
        $\widehat{Q} \leftarrow (\mathbf{q}_1,\dots,\mathbf{q}_{i-1},\widehat{\mathbf{q}}_i,\mathbf{q}_{i+1},\dots,\mathbf{q}_M)$, which is an M-removal
        if $\widehat{Q} \notin I$ and $\widehat{Q} \notin R$ **then** ▷ *NB: this means we have not visited vertex $\widehat{Q}$ yet.*
            Add $\widehat{Q}$ to I, i.e. $I \leftarrow I \cup \{\widehat{Q}\}$
        end if
    end for
    $R \leftarrow R \cup \{Q\}$
end while
Return R

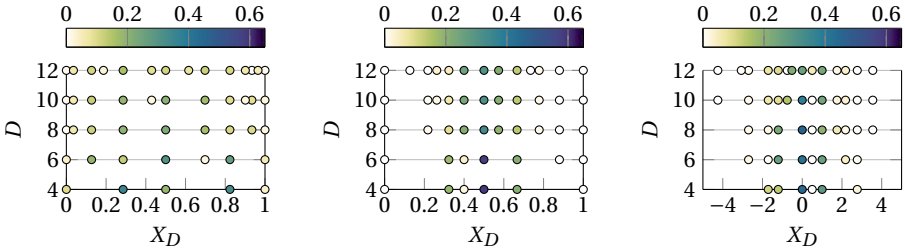
The set-up is similar to the one used so far, i.e. let $\{\mathbf{y}_k\}_{k=0}^{\infty}$ be a sequence of samples, $X_N^{(K)}$ be a set of nodes, and $W_N^{(K)}$ be a set of non-negative weights, and assume that the following holds for a certain D:

$$\sum_{k=0}^{N} \varphi_j(\mathbf{x}_k)\, w_k = \mu_j^{(K)}, \text{ for } j = 0,\dots,D,\ \mathbf{x}_k \in X_N^{(K)}, \text{ and } w_k \in W_N^{(K)}.$$

Let D_+ and K_+ be the desired number of basis vectors and (possibly larger) number of samples, respectively, and assume $D_+ \ge D$. The goal is to determine $X_{N+M}^{(K_+)}$ and $W_{N+M}^{(K_+)}$ such that $W_{N+M}^{(K_+)}$ is non-negative, $X_N^{(K)} \subset X_{N+M}^{(K_+)}$, and

$$\sum_{k=0}^{N+M} \varphi_j(\mathbf{x}_k)\, w_k = \mu_j^{(K_+)}, \text{ for } j = 0,\dots,D_+,\ \mathbf{x}_k \in X_{N+M}^{(K_+)}, \text{ and } w_k \in W_{N+M}^{(K_+)}.$$

In other words, we consider $K_+ + 1$ samples (denoted previously by Y_{K_+}) and want to



(a) $U[0,1]$ **(b)** $\beta(10,10)$ **(c)** $\mathcal{N}(0,1)$

Figure 4.6: Examples of nested nodal sets constructed with 10^5 samples. The initial nodes are in all three cases $[0, 1/2, 1]$. The colors indicate the weights of the nodes.

determine a positive quadrature rule that integrates all $\varphi \in \Phi_{D_+}$ exactly by adding M nodes to X_N (with M minimal).

The iterative procedure is similar to Algorithm 4.1 and consists of four steps: firstly determine or obtain the next sample $\mathbf{y}_{K+1}$, secondly update the nodes and weights according to (4.9), thirdly determine all possible removals (see Algorithm 4.2), and finally remove nodes such that the obtained quadrature rule is as small as possible. The last step consists of finding the M-removal such that $X_{N+M}^{(K_+)} \setminus X_N^{(K)}$ (i.e. the set of new nodes) becomes as small as possible. If a node $\mathbf{x}_k \in X_N^{(K)}$ is part of the M-removal, its weight is simply set to 0 (this is not problematic: the weights change again in subsequent iterations).

The initialization of the algorithm depends on whether more basis vectors are considered, a larger number of samples is considered, or the set of samples is changed (e.g. the sequence of samples is redrawn):

1. If $D_+ = D$ and $K_+ > K$, the procedure is a continuation of the original Algorithm 4.1 and no initialization is necessary.

2. If $D_+ > D$, we need to reiterate over all samples to determine $\mu_j^{(K)}$ for $j > D$. The algorithm can be initialized using $X_N^{(K)}$ as nodes, using all weights equal to $1/(N+1)$, and using the samples $Y_{K_+} \setminus X_D^{(K)}$.

3. If the sequence of samples is regenerated from the underlying distribution, then in general $X_N^{(K)} \not\subset Y_{K_+}$. Therefore, the algorithm is initialized with $X_N^{(K+1)} \cup \{\mathbf{y}_0\}$ and $W_N^{(K+1)} = \{0, \ldots, 0, 1\}$.

The outline of this algorithm is provided in Algorithm 4.3, which is a straightforward extension of Algorithm 4.1 with additional bookkeeping to incorporate the removal of multiple nodes. Some examples of nested sequences are gathered in Figure 4.6 and 4.7. In the first figure all three nodal sequences are initialized with the nodes 0, 1/2, and 1. If these nodes are used to construct conventional interpolatory quadrature rules, then the quadrature rule is positive if the uniform or Beta distribution is used, but has

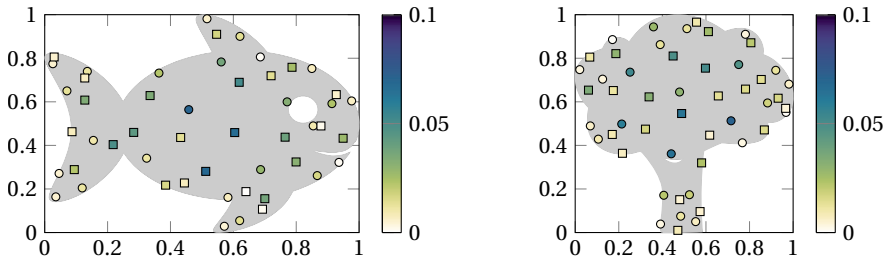


Figure 4.7: The implicit quadrature rule of 25 nodes (circles) and 50 nodes (circles and squares) respectively determined using the uniform distribution restricted to the gray area, using 10^5 samples. The colors refer to the weights of the *largest* quadrature rule.

negative weights if the normal distribution is considered (the weights are 3, −4, and 2 respectively). However, the proposed algorithm incorporates these nodes without difficulty in subsequent quadrature rules, resulting into positive weights. Note that the quadrature rules of polynomial degree 4 have six nodes in case of the Beta and normal distribution. The subsequent quadrature rules of the normal distribution have a higher number of nodes than the degree, which is due to the "bad" initial set of nodes.

A two-dimensional example is presented in Figure 4.7. Here, the initial quadrature rule is depicted with circles and the extension thereof with squares.

Again it holds that different sample sets produce different quadrature rules. Similarly as in Section 4.3.1, this can be eradicated by using deterministic samplers. An additional degree of freedom arises when choosing the M-removal, as there might be several M-removals that remove the largest number of nodes from $X_{N+M}^{(K_+)} \setminus X_N^{(K)}$. In the quadrature rules constructed in this thesis, we select one randomly.

The large advantage of the nested implicit quadrature rule is that it is dimension agnostic, basis agnostic, space agnostic, and distribution agnostic, which are properties that carry over from the basic implicit quadrature rule. Virtually any space and any distribution can be used, as long as the distribution has finite moments and a set of samples can be generated, can be determined, or is available.

4.4. Numerical examples

Two different types of test cases are employed to demonstrate the discussed properties of our proposed quadrature rule, in particular the independence from the underlying distribution.

The first class of cases consists of explicitly known test functions and distributions to assess the accuracy of the quadrature rule for integration purposes. To this end, the Genz integration test functions [66] are again employed and a comparison is made with a Monte Carlo approach. Moreover, it is instructive to see how the convergence compares with that of a sparse grid, although the comparison is strictly speaking incorrect, as a Smolyak sparse grid converges to the true integral.

Algorithm 4.3: The nested implicit quadrature rule

Input: Samples $\{\mathbf{y}_0, \dots, \mathbf{y}_{K_{\max}}\}$, quadrature nodes X_N, basis polynomials $\{\varphi_0, \dots, \varphi_D\}$
Output: Positive quadrature rule X_{N+M}, W_{N+M}

Initialize $X_N^{(N)}$ and $W_N^{(N)}$, e.g. $X_N^{(N)} \leftarrow X_N$, $W_N^{(N)} \leftarrow \{1/(N+1), \dots, 1/(N+1)\}$
$M \leftarrow 0$
for $K = N, \dots, K_{\max}$ **do**
  Add node:
  $X_{N+M+1}^{(K+1)} \leftarrow X_{N+M}^{(K)} \cup \{\mathbf{y}_{K-N}\}$
  $W_{N+M+1}^{(K+1)} \leftarrow (K+1)/(K+2) W_{N+M}^{(K)} \cup \{1/(K+2)\}$
  $M \leftarrow M+1$

  Update weights:
  Construct $V_D(X_{N+M}^{(K+1)})$
  Determine null vectors $\mathbf{c}^1, \dots, \mathbf{c}^M$ and determine all M-removals

  Choose:
  Let $Q = (\mathbf{q}_1, \dots, \mathbf{q}_M)$ be an M-removal
  ▷ *NB: we choose one that makes* $X_{N+M}^{(K+1)} \setminus X_N$ *the smallest.*

  Remove node:
  Let $(\alpha_1, \dots, \alpha_M)$ such that $w_k - \sum_{j=1}^M \alpha_j c_k^j = 0$ for all k with $\mathbf{x}_k \in Q$
  $\widehat{M} \leftarrow \#\left\{\mathbf{x}_k \in X_{N+M}^{(K+1)} \mid \mathbf{x}_k \notin X_N \text{ and } w_k - \sum_{j=1}^M \alpha_j c_k^j = 0\right\}$
  $X_{N+\widehat{M}}^{(K+1)} \leftarrow \left\{\mathbf{x}_k \in X_{N+M}^{(K+1)} \mid w_k - \sum_{j=1}^M \alpha_j c_k^j > 0 \text{ or } \mathbf{x}_k \in X_N\right\}$
  $W_{N+\widehat{M}}^{(K+1)} \leftarrow \left\{w_k - \sum_{j=1}^M \alpha_j c_k^j \mid \mathbf{x}_k \in X_{N+\widehat{M}}^{(K+1)}\right\}$
  $M \leftarrow \widehat{M}$
end for
Return $X_{N+M}^{(K_{\max})}$, $W_{N+M}^{(K_{\max})}$

Secondly, a partial differential equation with random coefficients is considered, where the goal is to infer statistical moments of the solution of a differential equation with random boundary conditions. The equations under consideration are the inviscid Euler equations modeling the flow around an airfoil, where the inflow parameters and the shape of the airfoil are assumed to be uncertain.

The Genz integration test functions are studied in Section 4.4.1. The Euler equations are considered in Section 4.4.2.

4.4.1. Genz test functions

The Genz integration test functions [66] are a set of test function to assess the accuracy of numerical integration routines, which have been used in Chapter 3 to assess the accuracy of the constructed univariate quadrature rules. In this chapter, the multivariate Genz test functions are considered and used to test the implicit quadrature rule, see Table 4.1.

Table 4.1: The multivariate test functions from Genz [66]. All d-variate functions depend on the d-element vectors $\mathbf{a}$ and $\mathbf{b}$. The vector $\mathbf{b}$ is an offset parameter to shift the function. The vector $\mathbf{a}$ describes the degree to which the family attribute is present.

Integrand Family	Attribute
$u_1(\mathbf{x}) = \cos\left(2\pi b_1 + \sum_{i=1}^d a_i x_i\right)$	Oscillatory
$u_2(\mathbf{x}) = \prod_{i=1}^d \left(a_i^{-2} + (x_i - b_i)^2\right)^{-1}$	Product Peak
$u_3(\mathbf{x}) = \left(1 + \sum_{i=1}^d a_i x_i\right)^{-(d+1)}$	Corner Peak
$u_4(\mathbf{x}) = \exp\left(-\sum_{i=1}^d a_i^2 (x_i - b_i)^2\right)$	Gaussian
$u_5(\mathbf{x}) = \exp\left(-\sum_{i=1}^d a_i \lvert x_i - b_i\rvert\right)$	C^0 function
$u_6(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 > b_1 \text{ or } x_2 > b_2 \\ \exp\left(\sum_{i=1}^d a_i x_i\right) & \text{otherwise} \end{cases}$	Discontinuous

The characteristics of these functions are similar to the univariate test functions, i.e. each has a certain attribute that is challenging for most numerical integration routines. The exact value of the integral of any of these test functions on the unit hypercube can be determined exactly. The goal is to assess the absolute integration error for increasing number of nodes in a five-dimensional setting, i.e. to assess

$$e_N = \left|\mathcal{A}_N^{(K_{\max})} u - \mathcal{I}^{(K_{\max})} u\right| = \left|\sum_{k=0}^{N} u(\mathbf{x}_k) w_k - \frac{1}{K_{\max}+1} \sum_{k=0}^{K_{\max}} u(\mathbf{y}_k)\right|,$$

for samples $\mathbf{y}_0, \ldots, \mathbf{y}_{K_{\max}}$ and various increasing N. The number of samples is chosen such that the quadrature error dominates and the sampling error $|\mathcal{I}^{(K_{\max})} u - \mathcal{I} u|$ is small. We compare the approximation with that of a Monte Carlo approach, where we assess the following error:

$$e_N = \left|\frac{1}{N+1} \sum_{k=0}^{N} u(\mathbf{y}_k) - \frac{1}{K_{\max}+1} \sum_{k=0}^{K_{\max}} u(\mathbf{y}_k)\right|,$$

i.e. it is considered as a quadrature rule with nodes $\{\mathbf{y}_k\}_{k=0}^N$ and weights $1/(N+1)$. If the underlying distribution can be decomposed as a product of univariate distributions (it is "tensorized"), we also study the Smolyak sparse grid, which is constructed using exponentially growing Clenshaw–Curtis quadrature rules in conjunction with the combination rule [135], as defined by (2.17) (see Section 2.1.3 and in particular Figure 2.6 on page 21). The sparse grid converges to the true value of the integral and therefore we use the true value of the integral to assess its convergence, even though the comparison is not completely fair in this case.

The numerical experiment is repeated twice for two different input distributions. First, the uniform distribution is used to be able to compare the methodology with conventional quadrature rule methods. Second, a highly correlated multivariate distribution (inspired by Rosenbrock function) is used to demonstrate the independence of the convergence rate from the input distribution.

To obtain meaningful results, the offset and shape parameters **a** and **b** of the Genz functions are chosen randomly and the numerical experiment is repeated 50 times. The obtained 50 absolute integration errors are averaged and the average is denoted by $\overline{e}_N$. The vector **a** is obtained by firstly sampling uniformly from $[0,1]^5$ and secondly scaling **a** such that $\|\mathbf{a}\|_2 = 5/2$. The vector **b** is uniformly distributed in $[0,1]^5$ without further scaling, as it is an offset parameter.

The implicit quadrature rule is generated with $K_{\max} = 10^4$ samples which are drawn randomly from the two input distributions respectively and the Monte Carlo approximation is determined using a subset of these samples, such that both the Monte Carlo approximation and the implicit quadrature rule converge to the same result. The initial quadrature rule of one single node is determined randomly and the rule is extended by applying Algorithm 4.3. Each extension is chosen such that D doubles, up to $D = 2^{10} = 1024$, but we emphasize that any granularity can be used here. Recall that $\Phi_D = \text{span}\{\varphi_0, \ldots, \varphi_D\}$, where φ_j are d-variate polynomials sorted using a graded reverse lexicographical order. Hence each extension integrates a larger number of polynomials exactly. For sake of completeness, a comparison is made with a non-nested implicit quadrature rule, which is regenerated for each D by means of Algorithm 4.1.

Uniform distribution

The multivariate uniform distribution in $[0,1]^d$ (with $d = 5$ in this case) can be constructed by means of a tensor product of multiple univariate uniform distributions. It is therefore possible to approximate the integral using the Smolyak sparse grid. The results of the four integration routines under consideration (Monte Carlo, nested and non-nested implicit quadrature rules, and Smolyak sparse grid) are depicted in Figure 4.8. Here, N denotes the number of nodes of the quadrature rules and the Smolyak sparse grid is refined by increasing the sparse grid level equally in all dimensions.

The accuracy of a quadrature rule is highly dependent on the analyticity and smoothness of the integrand. Globally analytic functions can be approximated well using polynomials, i.e. $\inf_{\varphi \in \Phi_D} \|\varphi - u\|_\infty$ decays fast. This property is reflected in the results.

The first four Genz functions (i.e. u_1, u_2, u_3, and u_4) are smooth and therefore the most suitable for integration by means of a quadrature rule. The best convergence is observed for the oscillatory, corner peak, and Gaussian function, which are analytic. The corner peak is analytic, but has very slowly decaying derivatives, such that the quadrature rule approximation only converges exponentially fast for very large numbers of nodes (which are not considered here).

The continuous (but not differentiable) C^0 function follows a similar reasoning. It is not globally analytic, hence no exponential convergence is obtained. The Smolyak quadrature rule has a slightly larger error in this case compared to the implicit quadrature rule (arguably due to its negative weights), even though it seems that the rate of convergence is similar.

Integrating the discontinuous function by means of a positive quadrature rule does not yield any improvement over Monte Carlo sampling. The Smolyak sparse grid performs worse in this case due to its negative weights and usage of the Clenshaw–Curtis quadrature rule (which is not suitable for integration of discontinuous functions).

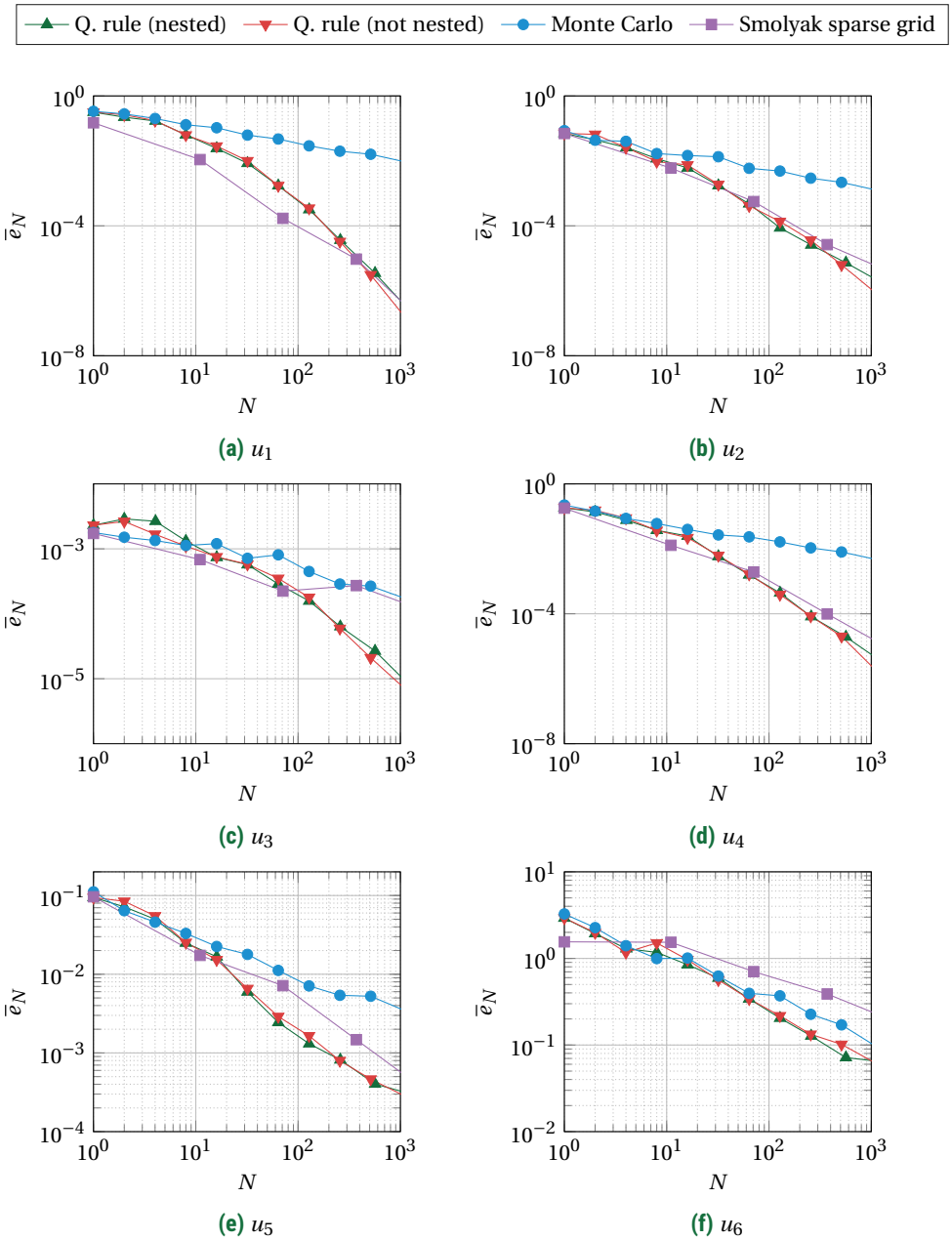


Figure 4.8: Convergence of the absolute integration error for Genz test functions using the nested and non-nested implicit quadrature rules, Monte Carlo sampling, and the Smolyak sparse grid using the uniform distribution.

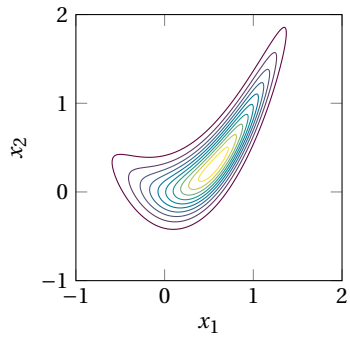


Figure 4.9: The bivariate Rosenbrock distribution.

Rosenbrock distribution

The large advantage of the implicit quadrature rule is that it can be constructed using any arbitrary set of samples. In order to assess this applicability to general distributions, the following distribution (which we will call the Rosenbrock distribution) is considered:

$$\rho\colon \mathbb{R}^d \to \mathbb{R}, \text{ defined by } \rho(\mathbf{x}) \propto \exp\left[-f(\mathbf{x})\right]\pi(\mathbf{x}),$$

with π the PDF of the multivariate standard Gaussian distribution and f (a variant of) the multivariate Rosenbrock function:

$$f(\mathbf{x}) = f(x_1, \ldots, x_d) = \sum_{i=1}^{d-1}\left[b\left(x_{i+1} - x_i^2\right)^2 + \left(a - x_i\right)^2\right], \text{ with } a = 1 \text{ and } b = 10.$$

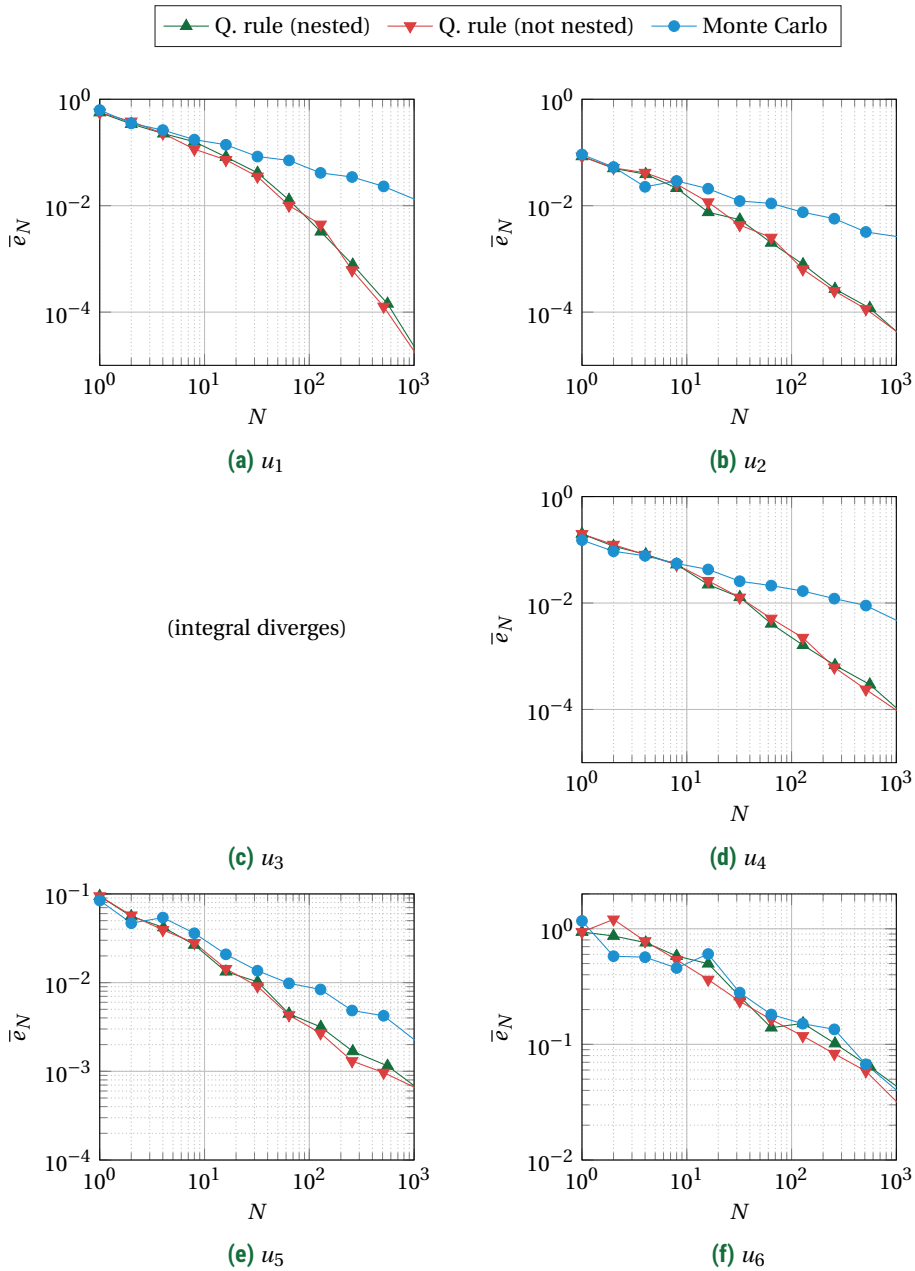
The distribution ρ for $d = 2$ is depicted in Figure 4.9. This distribution is not optimal for integration by means of a sparse grid as it cannot be decomposed in a product of univariate i.i.d. distributions. This means that integration by means of a sparse grid converges prohibitively slow, even if the quadrature rules used for the construction are based on the marginals of the distribution. Therefore these results are omitted.

The *exact* integral over the product peak function u_3 diverges in this case, so approximating such an integral will result in a diverging quadrature rule. The results of the other functions are gathered in Figure 4.10.

Similarly to the uniform case, the properties of the functions are reflected in the convergence rates of the approximations. The integrals of the smooth functions converge fast with a high rate, the convergence of the C^0 function is smaller, and the convergence of the discontinuous function is comparable to that of Monte Carlo. This result is significant, as it demonstrates that the convergence rate of the quadrature rule estimate to the sampling-based integral shows no significant dependence on the sample set used to construct the rules.

4.4.2. Airfoil flow using Euler equations

In this section the flow over an airfoil is considered with uncertain geometry and inflow conditions. The quantity of interest is the pressure coefficient of the airfoil. The problem



(a) u_1 (b) u_2 (c) u_3 (d) u_4 (e) u_5 (f) u_6

Figure 4.10: Convergence of the absolute integration error for Genz test function using the nested and non-nested implicit quadrature rules and Monte Carlo sampling using the Rosenbrock distribution.

Table 4.2: Uncertain parameters of the airfoil test case.

	Parameter	Distribution
α	Angle of attack	Uniform in $[0°, 5°]$
M	Mach number	Beta$(4,4)$ distributed in $[0.4, 0.6]$
t	Maximum thickness of the airfoil	Beta$(4,4)$ distributed in $[0.11, 0.13]$
m	Maximum camber	Beta$(2,\cdot)$ distributed in $[0.02, 0.03]$ with prescribed mean: $\overline{m} = 1/4\,t^2 + 0.02$
p	Location of maximum camber	Uniform in $[0.3, p_{\max}]$ with prescribed maximal value: $p_{\max} = \frac{1}{2}(2t + 16m)^5 + 0.3$

is five-dimensional: two parameters model uncertain environmental conditions and three parameters model the uncertain geometry of the airfoil. The geometry is described by the 4-digit NACA profile and the equations governing the flow are the inviscid Euler equations. Problems of this type are well known in the framework of uncertainty propagation [108, 110, 195] and allow us to demonstrate the applicability of the proposed quadrature rule to a complex uncertainty propagation test case in conjunction with a complex underlying distribution.

The five uncertain parameters are summarized in Table 4.2. The angle of attack and Mach number are distributed independently of the other parameters and describe uncertain inflow conditions. The remaining three parameters define the 4-digit NACA airfoil [88]. A NACA airfoil can be generated directly from these parameters and the mean of these parameters is approximately a NACA2312 airfoil. In this chapter the NACA airfoil with closed tip is considered by correction of the last parameter.

The compressible Euler equations are numerically solved using the finite volume solver `SU2` [50, 139]. The mesh is generated using `gmsh` [69]. The implicit quadrature rules are determined by means of $K_{\max} = 10^6$ randomly drawn samples from the distributions described in Table 4.2, that are also being reused for consecutive refinements.

The quantity of interest in this test case is the pressure coefficient on the surface of the airfoil $C_p(x)$, i.e. the scaled pressure such that zero pressure equals a non-obstructed flow:

$$C_p(x) = \frac{p(x) - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2}.$$

Here, $p(x)$ is the pressure at location x, p_∞ is the freestream pressure (i.e. the pressure on the boundary in this case), ρ_∞ is the freestream density of air, and V_∞ is the freestream velocity of the fluid. It is common to report the pressure coefficient as function of x, where $x \in [0,1]$ is a spatial parameter that runs from the leading edge to the trailing edge of the airfoil. For each location x there are two values of the pressure coefficient: one on the top and one on the bottom of the airfoil. Therefore, usually $-C_p$ as function of x is plotted, as in that case the line on the top refers to pressure on top of the airfoil (since the pressure on top of the airfoil is usually smaller) and vice versa.

The quadrature rule is applied piecewise to this quantity, where all pressure realizations are piecewise linearly interpolated onto the same mesh. Accuracy is measured

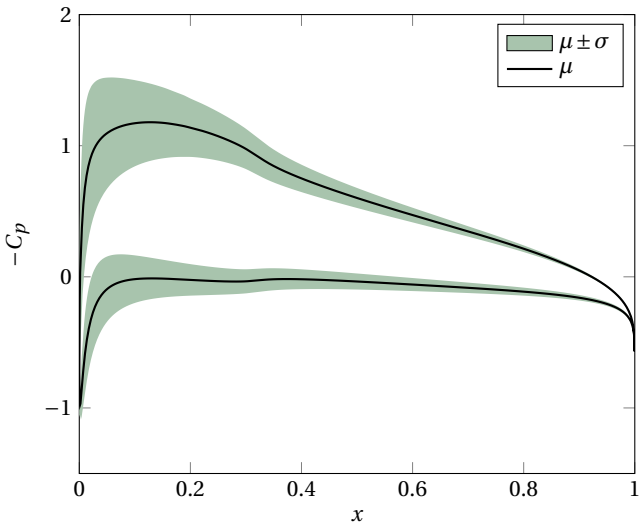


Figure 4.11: Mean μ and standard deviation σ of the pressure coefficient (C_p) on the airfoil determined using the finest quadrature rule.

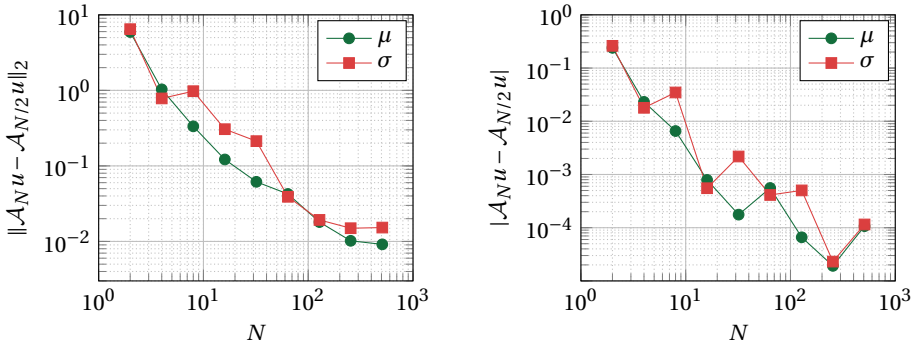
by using the lift coefficient C_l, which is the dimensionless coefficient relating the lift generated by the airfoil with the farfield fluid density and velocity. It follows naturally by integrating the pressure coefficient over the airfoil surface. In Figures 4.11, 4.12a, and 4.13 we used the quadrature rule $\mathcal{A}_N u$ with $u = C_p$ and in Figure 4.12b we used the quadrature rule with $u = C_l$.

The number of nodes of the quadrature rule is doubled until the difference between two consecutive quadrature rule estimations of the mean of the pressure coefficient is smaller than 10^{-2}, which is the case at 512 nodes. Here the difference is measured using the mean square error of the pressure coefficient at all grid locations on the airfoil (i.e. the 2-norm). The convergence of the pressure coefficient and the lift coefficient is depicted in Figure 4.12.

High order convergence is clearly visible. Both the mean and standard deviation show convergence with a larger rate than that of Monte Carlo (i.e. larger than 1/2). We want to emphasize the importance of positive weights for this engineering test case, as it ensures that the estimation of the variance is non-negative even in the presence of high non-linearities. The mean and standard deviation of the pressure coefficient on the airfoil, as determined using the quadrature rule of 512 nodes, are therefore well defined (see Figure 4.11).

The first four moments of the pressure coefficient around the airfoil are depicted in Figure 4.13, where the airfoil geometry that is plotted is the overlap of all airfoils (such that all depicted flow locations are in the flow for all quadrature rule nodes).

The largest uncertainty of the flow is near the leading edge of the airfoil. This is in contrast to higher Mach number flows, for which it has been observed that the region of largest uncertainty occurs near the shock wave [18, 195]. The latter type of flows are



(a) C_p **(b)** C_l

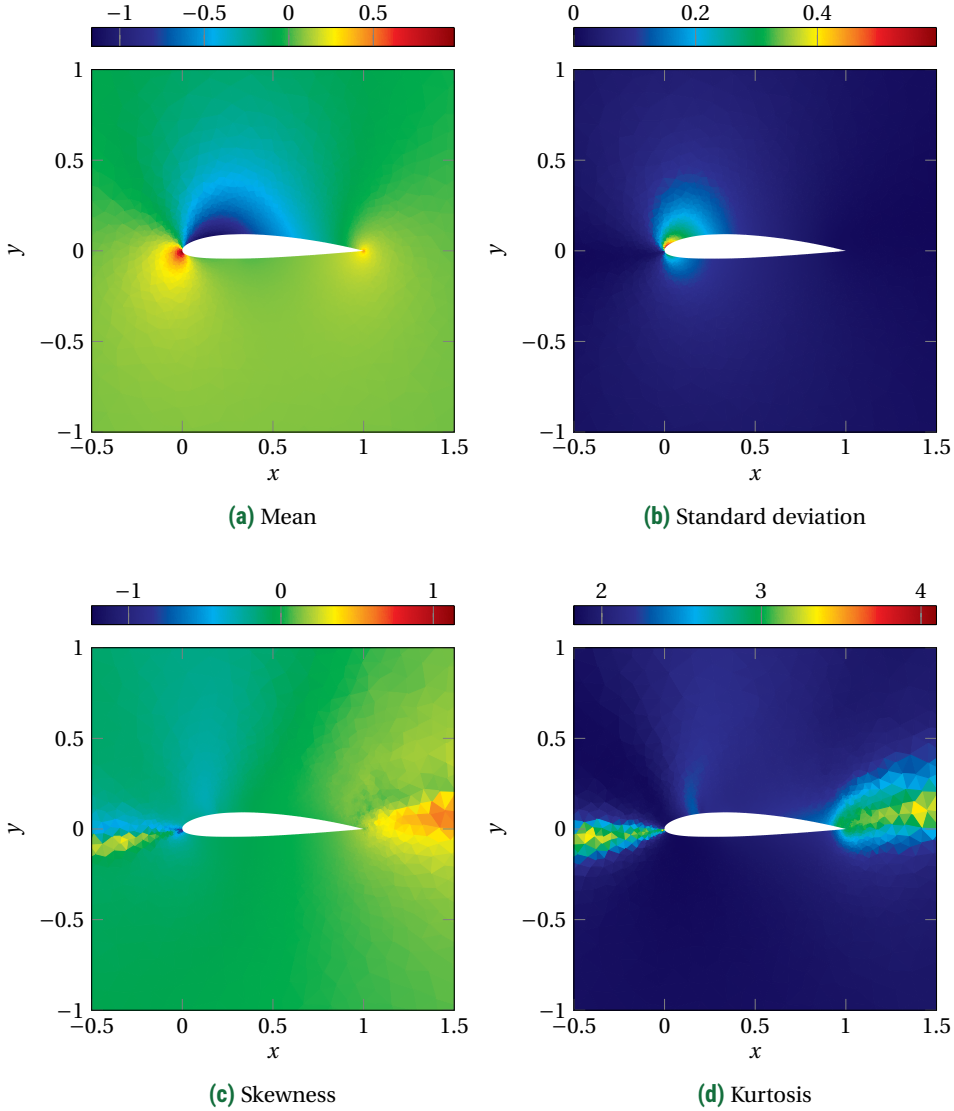
Figure 4.12: Convergence of the mean μ and standard deviation σ of the pressure coefficient C_p and lift coefficient C_l.

considered in Chapters 6 and 7. The skewness of the pressure coefficient shows that its distribution is slightly skewed near the stagnation point on the leading edge of the airfoil. In the wake of the airfoil the distribution has positive skewness, which means that outliers of the distribution will more likely be larger than smaller compared to the average pressure coefficient. The kurtosis demonstrates that in many regions of the flow the distribution has much less mass in the tails than a Gaussian (and is therefore more unlikely to produce outliers). However, again near the leading edge and trailing edge the distribution differs and the tails of the distributions are significantly more influential. We cannot conclude that these locations are the regions of highest uncertainty, as the standard deviation (which is the scaling factor of both the skewness and the kurtosis) is very small in these regions. It is merely a sign that the uncertain behavior of the flow cannot be fully captured by a Gaussian distribution.

4.5. Conclusion

In this chapter, a novel nested quadrature rule is proposed which is constructed by solely using samples from a distribution. It is called the *implicit quadrature rule*. The rule integrates an arbitrary high number of polynomials and the weights are positive, so high order convergence is obtained for sufficiently smooth functions. The algorithm to construct the quadrature rule ensures positive weights, high degree, and nesting regardless of the sample set. The quadrature rules are very suitable for the purpose of non-intrusive uncertainty propagation, because positive weights ensure numerical stability and nesting allows for refinements that reuse computationally expensive model evaluations.

The results from integrating Genz test functions demonstrate that the convergence rate of the quadrature rule is similar to that of the Smolyak sparse grid approach, if the underlying sample distribution is uncorrelated and defined on a hypercube. The real advantage of the proposed quadrature rule appears when this is not the case: for a correlated distribution on non-hypercube domains our method still converges at a rate



(a) Mean

(b) Standard deviation

(c) Skewness

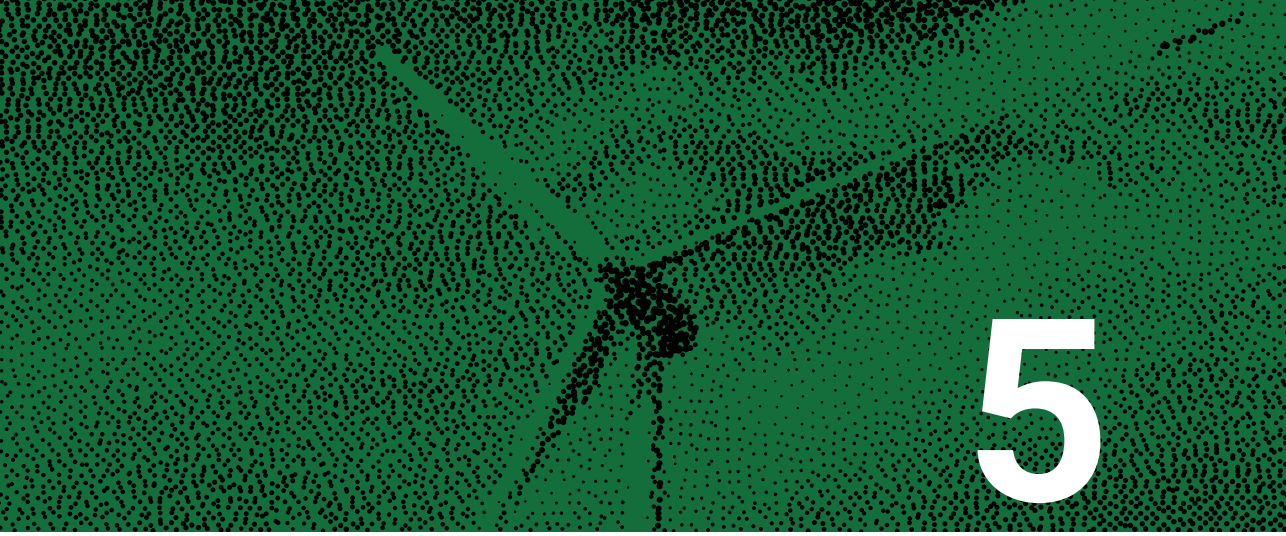
(d) Kurtosis

Figure 4.13: The first four centralized moments of the pressure coefficient around the airfoil.

similar to the uncorrelated case, while a sparse grid quadrature rule hardly converges at all. Similar to the existing quadrature rules, the convergence depends on the specific properties of the integrand, in particular on its smoothness.

To demonstrate the applicability to practical test cases, the implicit quadrature rule is used to determine the statistical moments of an airfoil flow with both independent and dependent input distributions. The results demonstrate the advantages of the quadrature rule: nesting can be used for easy refinements, positive weights ensure stability and positive approximations of positive quantities (such as the variance), dependency is naturally taken into account, and the accuracy of the rule yields high convergence rates.

The proposed algorithms provide a framework for the construction of quadrature rules that shows much potential for further extensions. For example, the rule as proposed in this chapter does not use any properties of the integrand. Tailoring the basis to the integrand can yield an adaptive quadrature rule without deteriorating the accuracy of the rule as a whole. As the rule is solely based on sample sets, no stringent assumptions are necessary to apply the quadrature rule in such a different setting. Other examples that are considered in this thesis include using the rule with actual measurement data as input, which is considered in Chapter 5, or using the rule in a Bayesian framework to create quadrature rules for posteriors, which is discussed in more detail in Chapter 7.



5 Fatigue design load cases using quadrature rule techniques

The implicit quadrature rule, as introduced in Chapter 4, forms a flexible approach to numerically approximate integrals with a computationally expensive integrand. It can be applied readily to standardized fatigue load cases, which are specific scenarios that describe situations a wind turbine has to withstand. Arguably one of the most computationally complex load case is considered in this chapter: the load case describing fatigue loading over the full life time of the turbine. The goal is to calculate equivalent loads acting on an offshore wind turbine such that the prescribed variability in the environmental conditions is incorporated. To facilitate this, the implicit quadrature rule is constructed using measurement data conducted at the North Sea. It showcases one of the main advantages of this numerical integration routine: no distributions have to be fitted to the measurement data, i.e. the quadrature rule *directly* incorporates the statistical moments of the measurements.

5.1. Introduction

For the certification of the design of a wind turbine, various load cases have been formulated in the IEC 61400 standard for both onshore [85] and offshore [86] wind turbines. These load cases vary from regular power production in commonly observed environmental conditions, to extreme conditions simulating storm and failure. All cases have one property in common: the aeroelastic simulation code that models the wind turbine must be evaluated a large number of times, which is computationally costly. In

This chapter is based on the following article: L. M. M. van den Bos, W. A. A. M. Bierbooms, A. Alexandre, B. Sanderse, and G. J. W. van Bussel. Fatigue design load calculations of the offshore NREL 5MW benchmark turbine using quadrature rule techniques. *To appear in Wind Energy*, 2019. arXiv: 1904.07021 [cs.CE].

The measurement data used in this chapter has been obtained as part of the Dutch Wind Op Zee project (www.windopzee.net) and has been provided by ECN part of TNO. Lindert Blonk and Menno Kloosterman from DNV GL are acknowledged for their technical support around BLADED and for providing the model of the NREL 5MW wind turbine.

this chapter arguably one of the most costly load cases is considered: Design Load Case (DLC) 1.2, describing fatigue loading for a power producing offshore wind turbine under regular environmental conditions. As the time span of this load case is the full lifetime of the turbine, usually a large number of model runs is necessary to assess the statistics of the fatigue loading.

The conventional approach to assess the effect of environmental variability on the turbine lifetime is to firstly split the domain of the variables describing the environmental conditions in bins, secondly run the aeroelastic model several times in each bin (the so-called *seeds*), and finally determine the quantity of interest (e.g. the weighted equivalent load) incorporating the probability of occurrence of each bin. This approach is suggested in the aforementioned standard [85, 86] and has been successfully applied in previous research [2, 62, 151].

If applicable, binning is a versatile and effective tool to assess the effect of parameter variability. However, a major disadvantage of binning is that the number of necessary evaluations of the aeroelastic model is often prohibitively large, which limits its applicability. Several approaches have been suggested over the years to alleviate the high computational cost, e.g. lumping as discussed in Kühn [98, Section 8.3]. This often requires non-trivial pre-processing steps and restrictive assumptions on the aeroelastic model. An approach that is related to lumping is based on specifically selecting scenarios that accurately represent the full load that acts on the turbine [172]. This approach performs best if only a small number of components is considered and if the loads acting on these components are strongly correlated. A different approach is to reduce the time of individual model runs by adaptively limiting the simulation time and extrapolating the obtained estimate [171], though this requires explicit insight in the used model. Many of these approaches can be combined straightforwardly with the methods discussed in this work, since all methods considered here are non-intrusive.

Since the goal of DLC 1.2 boils down to calculating an integral, various alternative approaches can be considered as a replacement of binning. Examples include for example quasi-Monte Carlo methods [31, 132], surrogate methods [4, 129, 201], or numerical integration techniques by means of quadrature approaches [15, 24, 46, 68, 116], as discussed extensively in the previous chapters. These approaches have in common that regularity in the model (i.e. continuous responses to variability in the input parameters) is leveraged such that fast convergence can be obtained for sufficiently smooth functions. Moreover, they have been successfully used to assess uncertainties in models in wind energy [20, 59, 75, 127, 128, 160], but a rigorous approach to model standardized load cases that directly competes with binning is still lacking. Furthermore, since the number of bins is fixed, it is difficult to assess the accuracy of the obtained estimation of the quantity of interest (the loads in the cases studied in this chapter). Contrary to the quadrature rules discussed in Chapter 4, it is often the case that explicit distributions of the parameters must be provided a priori. These are typically unavailable, as there are various distributions that could potentially fit the measurements of the uncertain environment [126].

The goal of this chapter is to use quadrature rule approaches to assess fatigue loads described by DLC 1.2, leading to a competitive and attractive alternative approach to binning. Quadrature rules have the advantage that they are tailored to calculating in-

tegrals, which is the key goal of the load case. Moreover, they are versatile and robust by providing fast convergence for smooth functions, while still yielding accurate results for non-smooth functions. The quadrature rule used in this chapter is the implicit quadrature rule, as discussed in Chapter 4. It has the additional advantage over conventional quadrature rules [46, 68, 116] that no explicit knowledge about the distribution of the environmental parameters is required: only measurements of the environmental conditions are necessary. Furthermore, it provides an error estimate of the estimated load.

This chapter is structured as follows. In Section 5.2 fatigue load calculation is introduced, consisting of describing DLC 1.2, the on-site conditions considered in this chapter, and the wind turbine under consideration. The propagation of parametric uncertainty through an aeroelastic model is discussed in Section 5.3. Here, the benchmark binning procedure is briefly explained and the quadrature rule algorithm and its most relevant mathematical properties are discussed in detail. The results obtained by binning and quadrature rules are compared in Section 5.4, demonstrating the applicability and prospect of quadrature rules. The chapter is concluded in Section 5.5.

5.2. Fatigue load calculation

The DLCs as specified in the design requirements for offshore wind turbines by IEC [85, 86] can be globally put into two categories: ultimate and fatigue load cases. The ultimate load cases describe the simulation of failure of a component due to rare events. Often these cases require non-trivial extrapolation procedures to assess the statistics. On the other hand, the fatigue load cases describe the simulation of regular environmental conditions over a long period of time to assess the effect of wear of the turbine. The last group is the focus of this chapter, as this type of cases requires a large number of evaluations of an aeroelastic code. In particular, the focus is on DLC 1.2 in combination with DLC 6.4. DLC 1.2 accounts for environmental conditions that result in power production and DLC 6.4 describes similar environmental conditions, with the sole exception that the turbine is idling due to too small or too large wind speeds*. There are various other fatigue load cases describing specific scenarios, e.g. start up (DLC 3.1), shutdown (DLC 4.1), and fault occurrence (DLC 2.4), but these are not the primary focus of this chapter since assessing these cases is significantly less computationally costly.

In this section, the specific details of the load case considered in this chapter are outlined. In Section 5.2.1 the details of the DLC are discussed, as standardized by IEC [85, 86]. In Section 5.2.2 and 5.2.3 the environmental conditions and the wind turbine under consideration are discussed. To keep the simulation as realistic and as reproducible as possible, freely available measurement data [190] and the NREL 5MW reference turbine [91] are used to describe the environment at the offshore site and model a wind turbine that should perform well at the location that is considered. In this thesis, the aeroelastic code `BLADED` is employed for the load calculations, which is briefly discussed in Section 5.2.4.

*Strictly speaking, DLC 6.4 only accounts for wind speeds that are significantly smaller than the reference wind speed of the turbine ($V_{\text{hub}} < 0.7\,V_{\text{ref}}$). We will also consider wind speeds larger than the cut-out speed, which also describe an idling wind turbine.

5.2.1. Design Load Case 1.2: fatigue analysis

The goal of DLC 1.2 and DLC 6.4 is to assess the effect of wear on a wind turbine over a long period of time under normal design situations. All conditions are equal for both load cases, except for the wind condition that describes an idling turbine in DLC 6.4. Basically DLC 6.4 covers the cases where the wind speed at hub height averaged over 10 minutes, denoted as V_{hub}, is smaller than the cut-in speed or larger than the cut-out speed and DLC 1.2 covers all other cases. It is convenient to bundle both cases in one single scenario and make no assumption about V_{hub}.

Given the mean wind speed, the turbulence intensity is defined by means of the *normal turbulence model* (NTM), which implies that the turbulence standard deviation σ_1 is given by the 90% quantile for the given wind speed at hub height, i.e.

$$\sigma_1 = I_{\text{ref}}\,(0.75\,V_{\text{hub}} + b), \text{ with } b = 5.6\,\text{m/s}.$$

The turbulence intensity, which should be used to generate an inflow wind field using the methods from Veers [184] or Mann [119], then equals $\sigma_1 / V_{\text{hub}}$. The lateral component of the turbulence standard deviation, denoted as σ_2, and the upward component, denoted as σ_3, are defined as $\sigma_2 = 0.8\sigma_1$ and $\sigma_3 = 0.5\sigma_1$ respectively (here, it is assumed that the Kaimal model is used). I_{ref} denotes the expected value of the turbulence intensity at 15 m/s, with $I_{\text{ref}} = 0.16$ for the case described in this chapter.

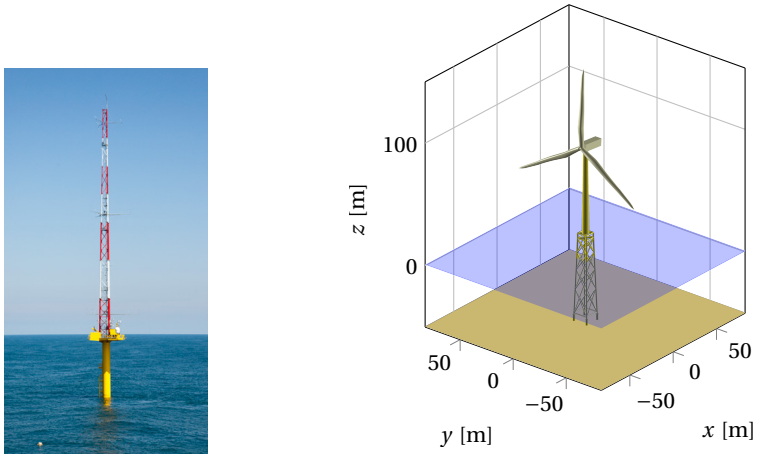
The wind direction θ_{wind} with respect to the turbine is assumed to be uniformly distributed between −12° and 12°, as the fatigue load case under consideration does not model extreme yaw misalignment. Hence the wind direction 0° is equivalent to no yaw misalignment. The wind direction and wind speed are independently distributed random variables.

The sea state is incorporated using a joint probability distribution of the significant wave height H_{s}, the peak spectral period T_{p}, and the wind at hub height V_{hub}. A normal sea state is considered, so no anomalous events are incorporated and currents are ignored. The wind and wave directionality (and their dependence) have to be incorporated as co-directional or multi-directional (i.e. they are either equal or not). In this chapter, the latter option is chosen. For this purpose, the wave direction θ_{wave} (with respect to the turbine) is used to determine the wind/wave misalignment:

$$\theta_{\text{wind/wave}} := \theta_{\text{wind}} - \theta_{\text{wave}}.$$

The distribution of the wind/wave misalignment can be determined by subtracting the stochastic wave direction (which should be inferred from the site conditions) from the uniformly distributed wind direction.

Combining the parameters describing the wind and sea conditions yields a five-dimensional parameter space, called Ω, consisting of all possible combinations of those parameters, called $\mathbf{x} \in \Omega$. It consists of the wind speed at hub height denoted by V_{hub}, the wind direction denoted by θ_{wind}, the significant wave height denoted by H_{s}, the peak spectral period denoted by T_{p}, and the wind/wave misalignment denoted by $\theta_{\text{wind/wave}}$. Hence $\mathbf{x} = (V_{\text{hub}}, \theta_{\text{wind}}, H_{\text{s}}, T_{\text{p}}, \theta_{\text{wind/wave}})^{\text{T}}$. The concrete goal of DLC 1.2 and DLC 6.4 is to assess the effect of regular, long-term variability of these parameters on the forces acting on the wind turbine.



(a) (b)

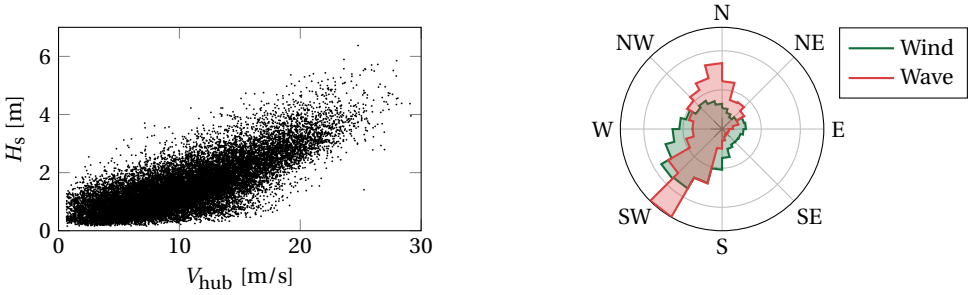
Figure 5.1: *Left:* The meteorological mast IJmuiden. *Right:* NREL 5MW reference wind turbine.

5.2.2. Meteorological mast IJmuiden measurements

The environmental conditions used for the load case calculations are based on the measurements done using meteorological mast IJmuiden [190]. This meteorological mast is situated approximately 85 km from the Dutch coast (at coordinates 52°50′53″N 3°26′8″E). At the measurement site the water depth is approximately 28 m, see Figure 5.1a for an illustration. The measurement campaign started in November 2011 and lasted until March 2016. Up to some interruptions, the wind conditions at various heights, wave conditions, and ocean current conditions at various depths have been obtained using a lidar and a buoy throughout that period of time.

For the purpose of this chapter, we are mainly interested in the statistical behavior of the wind in relation with the sea state. To this end, a straightforward pre-processing step is performed to combine the hourly measured sea state with the 10-minute averages of the wind conditions at that specific moment. After removal of all invalid measurements (due to measurement errors or interruptions), 24 650 samples of the wind speed and wind direction at hub height (which is 90 m, see the next section) and significant wave height, wave direction, and peak spectral period are obtained. These samples of the combined wind and wave conditions constitute 88% of all available measured wave directions (measured hourly) and 10% of all available measured wind directions (measured every 10 minutes).

The obtained measurements are visualized in Figure 5.2. In these figures the alignment of the wind turbine is ignored, i.e. the measured wind and wave direction are used to determine the wind/wave misalignment, which is consequently used for all computations. Notice that Figure 5.2a clearly shows that there are no measurements with $V_{\text{hub}} = 0\,\text{m/s}$, i.e. the measurement devices have a small tolerance resulting in a measured wind speed that is unconditionally positive. Figure 5.2b demonstrates that



(a) Mean wind speed and significant wave height

(b) Wind and wave direction

Figure 5.2: Measurements of the meteorological mast IJmuiden. *Left:* All measurements of the wind speed and significant wave height, demonstrating their mutual dependence. *Right:* Wind rose illustrating a histogram of the wind and wave direction. Both have a dominant south west inflow.

the dominant wind and wave direction is south west. There is also a significant number of waves from north, possibly due to the open connection with the North Sea at the offshore site of the meteorological mast. Nonetheless, waves from north rarely occur in combination with wind from south west, as the wind/wave misalignment has approximately zero mean.

5.2.3. NREL 5MW reference offshore wind turbine

The NREL 5MW reference wind turbine for offshore system development is a fictional offshore wind turbine designed to support concept studies aimed at assessing offshore wind technology [91]. It is a horizontal-axis wind turbine rated at 5 MW with a rotor diameter of 126 m. Its hub height is 90 m, which allows for the straightforward usage of the measurement data discussed in the previous section, as measurements at 90 m are available. The cut-in, rated, and cut-out wind speed of the turbine equal 3 m/s, 11.4 m/s, and 25 m/s respectively. As the mean wind speed of the meteorological mast at 90 m is 10.13 m/s, simulation of the NREL 5MW wind turbine at the location of the IJmuiden mast describes a realistic scenario of a power producing wind turbine. The geometry of the turbine is depicted in Figure 5.1b.

5.2.4. Aeroelastic code BLADED

The simulation of the turbine is performed using the aeroelastic wind turbine modeling tool `BLADED` (version 4.6). One run of `BLADED` to obtain the forces acting on the turbine over a period of 10 minutes takes approximately 50 minutes on the hardware used for the numerical experiments, which significantly dominates the total computation time. All other algorithms discussed in this chapter (such as determining the bins, computing the quadrature rules, and all post-processing procedures) are negligible in this regard.

`BLADED` is a simulation tool for modeling various aspects of a wind turbine. The main component used in this chapter is the module that models a power producing turbine in a regularly occurring environment. This module yields the time history of

the forces acting on various components of the turbine, given the time history of the environmental conditions. The wind turbine is modeled in BLADED using a combination of blade element momentum theory and mass-spring-damper models. Many advanced corrections, which might be based on physical insight, are used to ensure that the obtained results are accurate. It is out of the scope of this chapter to fully discuss all details. The interested reader is referred to Burton et al. [28] and the references therein for a general introduction to wind turbine modeling or the Theory Manual of BLADED [12] for the specific implementation details within BLADED.

Given specific values of the environmental parameters, BLADED is used firstly for simulating the wind turbine to obtain the forces and secondly for post-processing the obtained results. Obtaining forces consists of two steps. Firstly, a turbulent wind field using the average wind speed at hub height and the turbulence intensity is generated. Then secondly, using this turbulent wind field, the random sea state, and the design of the NREL 5MW wind turbine, a time history of the forces acting on various components of the turbine is calculated. By repeatedly doing such calculations for various environmental conditions, and incorporating the frequency of occurrence of each specific set of environmental parameter values, weighted equivalent loads are calculated for various representative slopes of the S–N curve. An equivalent load is a representative load of a specific load time history resulting in the same damage. A weighted equivalent load is an equivalent load where the frequency of occurrence of several time histories is taken into account. Determining these loads is the primary goal of this chapter and these loads form a standardized way of reporting and assessing wind turbine performance.

There exists variability in the output of BLADED due to the random nature of the turbulent wind field and sea state. To model these, it is common to generate several wind fields and sea states and average the time histories of the forces over these realizations. These so-called *seeds* are also incorporated in the framework in this chapter.

It is important to emphasize that the procedure of determining equivalent loads and calculating the seeds is similar for both binning, which is the well-known conventional approach, and numerical integration by means of quadrature methods, which is the novel approach requiring significantly less BLADED evaluations suggested in this chapter. No modifications to the code or input files of BLADED are necessary, except for the frequency of occurrence of each individual simulation. This is a major advantage of the approach suggested in this chapter, as all existing well-developed frameworks can be used without significant changes (i.e. the approach is *non-intrusive*).

5.3. Numerical integration for load calculations

The procedure to determine all equivalent loads that are necessary to obtain weighted equivalent loads is the most costly part of DLC 1.2, as it requires the calculation of a large number of time histories of loads. In this section, firstly this effect is quantified and secondly the load case is embedded in a numerical integration framework to facilitate the application of quadrature rule techniques.

To introduce equivalent loads using the mathematical nomenclature of this thesis, let $u\colon \Omega \to \mathbb{R}$ be the map that yields an equivalent load (the quantity of interest) for specific values of the environmental conditions $\mathbf{x} \in \Omega$ (see Section 5.2.1 and the nomenclature of Chapter 3 and 4). To incorporate the uncertain nature of the environment, the equivalent

loads are combined into a single weighted equivalent load, which is ideally calculated as follows:

$$L^{\mathrm{eq}} = \sqrt[m]{\mathbb{E}[u(\mathbf{X})^m]}, \text{ with } \mathbb{E}[u(\mathbf{X})] = \int_\Omega u(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{5.1}$$

Here, $\rho(\mathbf{x})$ is the probability density function of the random variable $\mathbf{X}$, that models the uncertainty of the environment. The integer m is the slope of the S–N curve, where it is implicitly assumed that the S–N curve can be represented by a monomial with power m. The notation L^{eq} is used to represent this (ideal) weighted equivalent load.

In the case studied in this chapter, it is not assumed that there is a known probability density function of the environmental parameters, but instead measurements are directly used (see Section 5.2.2). In this case, the random variable $\mathbf{X}$ is discrete (as discussed in Section 4.2) and L^{eq} can be introduced as follows:

$$L^{\mathrm{eq}} = \sqrt[m]{\mathbb{E}[u(\mathbf{X})^m]}, \text{ with } \mathbb{E}[u(\mathbf{X})] = \frac{1}{K}\sum_{k=0}^{K-1} u(\mathbf{y}_k), \tag{5.2}$$

where $\mathbf{y}_k \in \Omega$ are all measurement points available (as depicted in Figure 5.2) and $K = 24\,650$ depicts the number of measurement points. Each $\mathbf{y}_k$ is a five-dimensional vector describing the measured parameters (see Section 5.2.2). In theory, both (5.1) and (5.2) can be used in the framework proposed here, i.e. either measurement data or a distribution can be used straightforwardly.

Usually it is not viable to evaluate (5.1) exactly, as u is a map that involves evaluating a complex aeroelastic model. Also (5.2) cannot be used, as it is intractable to evaluate the aeroelastic model for each measurement point. Therefore, the integral is replaced by a quadrature rule, obtaining the following approximation:

$$L_N^{\mathrm{eq}} = \sqrt[m]{\mathcal{A}_N[u^m]}, \text{ with } \mathcal{A}_N u = \sum_{k=0}^{N} u(\mathbf{x}_k)\,w_k. \tag{5.3}$$

Here, $\mathbf{x}_0, \ldots, \mathbf{x}_N$ are $N+1$ specific representative values of the environmental parameters, forming the *nodes* of a quadrature rule in this chapter, and $w_0, \ldots, w_N$ are the *weights* accompanying the nodes. The weights model the frequency of occurrence of the nodes and therefore it is essential that these weights are positive. The notation L_N^{eq} is used to represent a weighted equivalent load determined using the quadrature rule $\mathcal{A}_N$. Ideally, we would have that $L_N^{\mathrm{eq}} \to L^{\mathrm{eq}}$ for $N \to \infty$ if L^{eq} is according to (5.1) or for $N \to (K-1)$ if L^{eq} is according to (5.2).

The standardized approach to determine the nodes and weights is by means of binning, which fits naturally in (5.3). In that case, $\mathbf{x}_k$ are equidistantly spaced and w_k equal the frequency of occurrence of each environmental condition. This approach is discussed briefly in Section 5.3.1.

One major disadvantage of binning is the large number of nodes (and concomitant model runs) that are obtained for the five-dimensional parameter space considered in DLC 1.2. An alternative is to determine the nodes and weights such that they form an interpolatory quadrature rule. This yields an accurate estimation with relatively small number of nodes. The challenge is to construct the rule that incorporates the measurement data without reconstructing a (possibly inaccurate) probability density

function and without requiring model evaluations at all samples $\mathbf{y}_k$. For this purpose, the implicit quadrature rule is employed, as discussed extensively in Chapter 4. The rule used for the load calculations, including its embedding in the framework of this chapter is discussed in Section 5.3.2.

In (5.1), (5.2), and (5.3) it is assumed that the function u can be evaluated exactly. This is generally not the case, as the obtained equivalent loads still depend on the generated wind field and random sea. This is often solved by using *seeds*, which consists of constructing various wind and sea states with similar environmental parameters and averaging the obtained equivalent loads. Since the obtained equivalent loads are used in a weighted sum, it is a natural idea to use more seeds for nodes with high weight and vice versa. With this idea, seeds can be incorporated in the quadrature rule framework. Details are discussed in Section 5.3.3.

5.3.1. Binning

Binning is the conventional approach to assess variability in the wind and waves. The idea is to evaluate the aeroelastic code at equidistantly placed locations and post-process the results using the probability density function of the variables under consideration. The bins can be interpreted as quadrature rule nodes, yielding a rule with equidistant $\mathbf{x}_k$. The weights w_k describe the frequency of occurrence of $\mathbf{x}_k$.

If a distribution $\rho(\mathbf{x})$ is known, the weight of the k-th bin b_k is typically determined as follows [85, Annex G]:

$$w_k = \int_{b_k} \rho(\mathbf{x})\,\mathrm{d}\mathbf{x}.$$

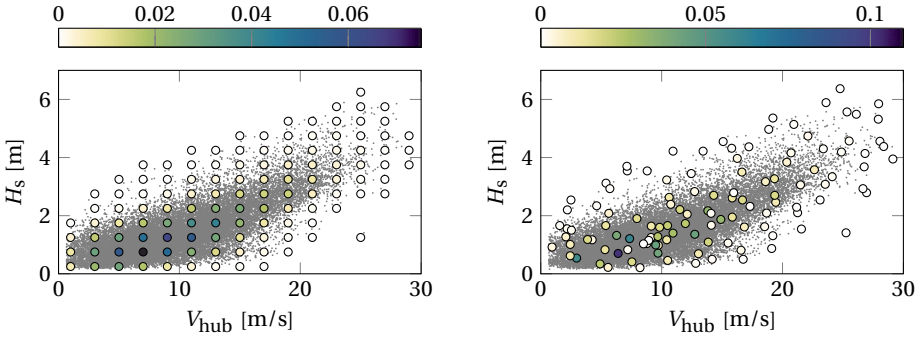
If measurements are considered, the weights are the fraction of measurements in the bin of $\mathbf{x}_k$, i.e.

$$w_k = \frac{K_k}{K},$$

where K_k is the number of measurements in the bin of $\mathbf{x}_k$ and K is the total number of measurements.

An example of a quadrature rule obtained by applying binning is illustrated in Figure 5.3a, where the same measurement points as in Figure 5.2a have been used to determine the bins. The weights are illustrated by means of colors and nodes with zero weight are not depicted. The mean wind speed at hub height is binned with bin sizes of 2 m/s and the significant wave height is binned with bin sizes of 0.5 m. Notice that many weights equal zero, requiring no concomitant model runs. Nonetheless, the majority of the nodes (117 out of 210) has possibly small, but positive weight.

The advantage of this approach is that it is versatile and straightforward to implement. Moreover it is supported by many software packages for wind turbine simulation. However, if multiple environmental parameters are considered the number of bins increases exponentially. To see this, notice that if d parameters are considered with B bins in each direction, the total number of bins is proportional to B^d, even if bins with zero weight are neglected. This severely limits the applicability of binning if a large number of parameters is considered.



(a) Binning **(b)** Quadrature rule

Figure 5.3: The representative values of the wind speed at hub height and significant wave height if binned or if determined by means of a quadrature rule. The color of the nodes indicates their weight.

5.3.2. Interpolatory quadrature rules

As an alternative to binning, we consider interpolatory quadrature rules, or more specifically, the implicit quadrature rule as introduced in Chapter 4. In this chapter, the fixed implicit quadrature rule as introduced in Section 4.3.1 is used, which is an interpolatory quadrature rule with positive weights. Subsequently a sequence of nested quadrature rules is constructed by iteratively removing nodes from this rule, as discussed in Section 3.3. This yields a sequence of interpolatory quadrature rules with positive weights.

Recall that the key property of an interpolatory quadrature rule is that it is constructed such that the number of polynomials it integrates exactly is equal to its number of nodes. For this purpose, reconsider $\Phi_N = \text{span}\{\varphi_0, \ldots, \varphi_N\}$, where $\{\varphi_k\}$ are monomials which are sorted graded lexicographically in this chapter. Then if $\mathbf{x}_0, \ldots, \mathbf{x}_N$ and $w_0, \ldots, w_N$ are the nodes and weights of an interpolatory quadrature rule, we have the following:

$$\mathcal{A}_N \varphi_j = \sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k = \mu_j, \text{ with } j = 0, \ldots, N.$$

The values μ_j are the raw moments of the distribution, defined as follows:

$$\mu_j = \mathbb{E}[\varphi_j] = \int_\Omega \varphi_j(\mathbf{x}) \rho(\mathbf{x}) \, \mathrm{d}\mathbf{x}, \text{ for } j = 0, \ldots, N,$$

which uses (5.1). If measurements are used, the raw moments are defined as follows:

$$\mu_j = \frac{1}{K} \sum_{k=0}^{K-1} \varphi_j(\mathbf{y}_k), \text{ for } j = 0, \ldots, N,$$

where $\mathbf{y}_k$ constitute all available measurements. The latter definition, which is considered in this chapter, uses (5.2).

Given all available measurements $\mathbf{y}_0, \ldots, \mathbf{y}_{K-1}$, the implicit quadrature rule prescribes that there exist nodes $X_N = \{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$ with $X_N \subset Y_K$ and positive weights $w_0, \ldots, w_N$ such that

$$\sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k = \frac{1}{K} \sum_{k=0}^{K-1} \varphi_j(\mathbf{y}_k), \text{ for } j = 0, \ldots, N.$$

Such nodes and weights can be computed using Algorithm 4.1 (see page 79).

The accuracy of the quadrature rule can be assessed by using the Lebesgue inequality (see (2.12) on page 16 and (4.5) on page 72). It yields that if the computational model *can* be approximated by means of a polynomial, a quadrature rule is a viable numerical integration methodology. A model can be approximated by a means of a polynomial if it is continuous and if its domain is bounded, which is the case for `BLADED` and the parameter space Ω considered in this chapter.

A major advantage of binning and quadrature rules is that both methods are non-intrusive. The nodes of the quadrature rules and the bins obtained after binning are independent of the model under consideration. This is especially relevant for load calculations, since it allows to compute the loads acting on all components using the same set of model evaluations. Usually it is not necessary to recompute the loads for each component. For instance, `BLADED` (and many other aeroelastic computer models) computes the loads on all components in one single run.

As discussed before, there exist various other approaches to construct interpolatory quadrature rules with positive weights. For example, well-known univariate quadrature rules include the Gaussian quadrature rules [74] and the Clenshaw–Curtis quadrature rules [35]. Higher dimensional quadrature rules often have similar limitations as binning: the number of nodes grows exponentially as the number of parameters grows. To alleviate this, the Smolyak sparse grid can be used [135, 169], as considered briefly in Section 2.1.3. However, in general the Smolyak sparse grid does not have positive weights, which could result in physically impossible estimates. Moreover, the Smolyak sparse grid requires that all parameters are independently distributed, which is clearly not the case for DLC 1.2, as described in Section 5.2.1.

An example of the implicit quadrature rule, determined using the measurement points from Figure 5.2a, is depicted in Figure 5.3b. The rule is chosen to consist of 117 nodes (the number of bins with non-zero weight). It is clearly visible that the nodes have positive weights and to a certain extent represent the distribution of the measurement points.

To assess the accuracy of the generated implicit quadrature rule, the removal procedure outlined in Section 3.3 is used. Given a quadrature rule, iteratively a function φ_j is removed (the last row of the Vandermonde matrix, this decreases the degree of the rule) and subsequently a node is removed from the quadrature rule using Carathéodory's theorem (see page 33). Consequently a *sequence* of quadrature rules with positive weights of decreasing accuracy and decreasing number of nodes is obtained, such that the following expression can be used to estimate the accuracy of the rules:

$$e_n = |\mathcal{A}_n u - \mathcal{A}_N u|.$$

Here, n denotes the number of nodes of the rule upon removal of $(N - n)$ nodes. Determining e_n for several increasing n (with $n < N$) gives insight in the decay of the error

made by the quadrature rule, as described by (4.7) (see page 73):

$$\Big||\mathcal{A}_n u - \mathbb{E}[u]| - |\mathcal{A}_N u - \mathbb{E}[u]|\Big| \leq e_n \leq \Big||\mathcal{A}_n u - \mathbb{E}[u]| + |\mathcal{A}_N u - \mathbb{E}[u]|\Big|.$$

The left-hand side of the inequality describes that e_n is larger than the difference of the errors made by $\mathcal{A}_n$ and $\mathcal{A}_N$. The right-hand side of the inequality describes that e_n is smaller than the sum of both errors. If the quadrature rule is applied to a function u that can be numerically integrated using a quadrature rule, a small e_n implies that the error of the quadrature rule is small. An even better estimation of the error can be obtained by repeatedly determining e_n for different sequences of quadrature rules, and taking the average of the obtained values. This is possible as multiple sequences exist given the freedom in choosing a node that is removed from the quadrature rule (based on the non-uniqueness of the null vector of the Vandermonde matrix).

5.3.3. Seed balancing

Both binning and quadrature rule approaches yield nodes that describe values of the environmental parameters that should be evaluated using the aeroelastic code. These evaluations have been denoted by the function $u(\mathbf{x})$. However, these evaluations still contain a degree of randomness: for each set of parameters, a certain number of *seeds* must be evaluated. This involves three steps: firstly various turbulent wind fields and sea states are constructed, secondly the loads are determined for *each* wind field and sea state, and finally the obtained loads are averaged. Hence, if S_k seeds are used to determine $u(\mathbf{x}_k)$, this can be denoted as follows:

$$u(\mathbf{x}_k) = u_{S_k}(\mathbf{x}_k) := \frac{1}{S_k}\sum_{s=1}^{S_k} u^{(s)}(\mathbf{x}_k),$$

where $u^{(s)}(\mathbf{x}_k)$ is an evaluation of the aeroelastic code using the s-th wind field and s-th sea state. Ideally, we would like to accurately approximate $\mathbb{E}[u_\infty]$, with

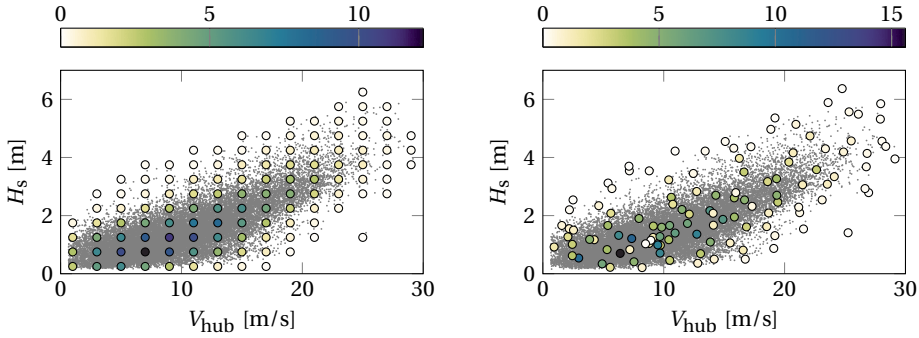
$$u_\infty(\mathbf{x}) = \lim_{S\to\infty}\frac{1}{S}\sum_{s=1}^{S} u^{(s)}(\mathbf{x}).$$

The approximation used in this chapter is $\mathcal{A}_N u_S$ for finite N and S. Here, $S = \sum_k S_k$ denotes the total number of `BLADED` evaluations used in this quadrature rule estimate. It yields the following error estimate[†]:

$$|\mathbb{E}[u_\infty] - \mathcal{A}_N u_S| \leq \underbrace{|\mathbb{E}[u_\infty] - \mathcal{A}_N u_\infty|}_{\text{Quadrature error}} + \underbrace{|\mathcal{A}_N u_\infty - \mathcal{A}_N u_S|}_{\text{Seed error}}.$$

The focus in this section is on the seed error, since the quadrature error has been discussed extensively in Section 5.3.2. It is common to use a fixed number of seeds

[†]Contrary to the separation of error that is conducted in Chapter 4, this expression does not contain a *sampling error*. It is assumed in this chapter that the measurements fully represent the environmental conditions, such that no sampling error is present. Nevertheless, incorporating it is straightforward, since it is independent of both the number of nodes and the number of seeds.



(a) Binning (323 seeds) **(b)** Quadrature rule (341 seeds)

Figure 5.4: Number of seeds per node if the seeds are balanced with an accuracy goal $\mathcal{E} = 1/\sqrt{5}$. The color of the nodes indicates the number of seeds (before rounding up to the nearest integer).

for each node (i.e. all S_k are equal). However, an error of similar order of magnitude determined with a smaller total number of seeds can be obtained by selecting the number of seeds per node based on the frequency of occurrence of the node. In this way, the accuracy in the overall approximation of the integral is preserved with a reduced cost. In this section this intuition is derived mathematically.

First, notice that the seed error behaves as follows [31]:

$$|\mathcal{A}_N u_\infty - \mathcal{A}_N u_S| \leq \sum_{k=0}^{N} |u_\infty(\mathbf{x}_k) - u_{S_k}(\mathbf{x}_k)| w_k, \text{ with } |u_\infty(\mathbf{x}_k) - u_{S_k}(\mathbf{x}_k)| \sim \frac{1}{\sqrt{S_k}}. \tag{5.4}$$

Throughout this section we write $A \sim B$ if the growth or decay of A is asymptotically the same as B. The constant of proportionality is the variance of the integrand, which typically depends on $\mathbf{x}_k$ and m (the inverse S–N slope, recall that the integrand in (5.3) is u^m). It is omitted from this estimation, but can straightforwardly be incorporated in the procedure discussed in this section, provided that its dependence on $\mathbf{x}_k$ and m is explicitly known.

Notice that (5.4) is true regardless of the exact model under consideration. The error behavior described by $1/\sqrt{S_k}$ follows from the usage of seeds and not from the usage of an inaccurate model. Hence a different model can be used without having to alter the seeds. As mentioned before, this is especially advantageous for load calculations: usually the loads on all components of the turbine are computed in one single model run and the proposed method discussed here can therefore be used to assess all equivalent loads without requiring extra calculations.

The computational time of `BLADED` does not depend on the specific wind field or sea state under consideration, so the total computational time scales linearly with the number of seeds. Hence given an accuracy goal $\mathcal{E}$, the goal is to determine $\varepsilon_{S_0}, \ldots, \varepsilon_{S_N}$ that firstly minimize the total computational time S, denoted as the total number of seeds used for all calculations, and secondly achieve the accuracy goal, denoted by the

5

Table 5.1: Example of reduction in number of seeds when applying seed balancing. The number of nodes equals 117 for both binning and the implicit quadrature rule. The accuracy goal $\mathcal{E}$ equals $1/\sqrt{5}$.

	Unbalanced	Balanced
Binning	585	323 (56 %)
Quadrature rule	585	341 (58 %)

seed error:

$$\text{Minimize } S = \sum_{k=0}^{N} S_k \sim \sum_{k=0}^{N} \frac{1}{\varepsilon_{S_k}^2}, \text{ with } \varepsilon_{S_k} := |u_\infty(\mathbf{x}_k) - u_{S_k}(\mathbf{x}_k)|, \text{ such that } \mathcal{E} \geq \sum_{k=0}^{N} \varepsilon_{S_k} w_k. \tag{5.5}$$

Notice that for $N \to \infty$, the asymptotic error of this expression decays to 0. Even though the error estimate is asymptotic, the relation $S_k = \varepsilon_{S_k}^{-2}$ will be used throughout this chapter.

The problem sketched in (5.5) is a constrained minimization problem, that can be solved using the method of Lagrange multipliers [8], which yields the solution

$$\varepsilon_{S_k} = A\left(\frac{w_k}{2}\right)^{-1/3},$$

where A is a scaling constant such that $\sum_{k=0}^{N} \varepsilon_{S_k} w_k = \mathcal{E}$. Hence the number of seeds for the k-th node should ideally equal

$$S_k = \frac{1}{\varepsilon_{S_k}^2} = A' w_k^{2/3}, \text{ with } A' = \frac{1}{2}\sqrt[3]{2}\, A^{-2} \approx 0.630\, A^{-2}. \tag{5.6}$$

The number of seeds S_k is larger if the associated weight w_k is larger and no seeds (hence no `BLADED` runs) are necessary if w_k is zero. The latter is straightforward to observe: if the weight of a node is zero, it does not contribute to the quadrature rule approximation. Notice that the relation between S_k and w_k is non-linear due to the non-linear relation between ε_{S_k} and S_k.

Only in rare cases it holds that all values of S_k as defined by (5.6) are integers. It is difficult to solve the optimization problem stated here with the restriction that all S_k are integer, so in this chapter the number of seeds is simply rounded up such that the accuracy goal is reached in all cases.

The effect of the seeds and their dependence on the weights is illustrated in Figure 5.4 and summarized in Table 5.1, where the quadrature rules from Figure 5.3 are reconsidered. Here it is assumed that by default five seeds are used per bin or node, which is equivalent to an accuracy goal of $\mathcal{E} = 1/\sqrt{5}$. If binning is used, the total number of seeds reduces from 585 (five per bin with non-zero weight) to 323, which is only 56% of the original cost. If the implicit quadrature rule from Section 5.3.2 is used, the total number of seeds reduces to 341, which is 58% of the original cost.

Table 5.2: The test functions from Genz [66]. All d-variate functions depend on the d-element vectors $\mathbf{a}$ and $\mathbf{b}$. The vector $\mathbf{b}$ is an offset parameter to shift the function. The vector $\mathbf{a}$ describes the degree to which the family attribute is present.

Integrand Family	Attribute
$u_1(\mathbf{x}) = \cos\left(2\pi b_1 + \sum_{i=1}^{d} a_i x_i\right)$	Oscillatory
$u_2(\mathbf{x}) = \prod_{i=1}^{d} \left(a_i^{-2} + (x_i - b_i)^2\right)^{-1}$	Product Peak
$u_3(\mathbf{x}) = \left(1 + \sum_{i=1}^{d} a_i x_i\right)^{-(d+1)}$	Corner Peak
$u_4(\mathbf{x}) = \exp\left(-\sum_{i=1}^{d} a_i^2 (x_i - b_i)^2\right)$	Gaussian
$u_5(\mathbf{x}) = \exp\left(-\sum_{i=1}^{d} a_i \lvert x_i - b_i \rvert\right)$	C^0 function
$u_6(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 > b_1 \text{ or } x_2 > b_2 \\ \exp\left(\sum_{i=1}^{d} a_i x_i\right) & \text{otherwise} \end{cases}$	Discontinuous

5.4. Numerical examples

In order to demonstrate the benefits of the proposed quadrature rule framework, the five-dimensional DLC 1.2 is considered and the approach is compared to binning. However, it is infeasible to assess the complete load case using binning, as it results in an excessive number of runs of `BLADED`. Therefore, first two simplified test cases are considered for the comparison. The first test case, which is discussed in Section 5.4.1, consists of assessing the accuracy of the quadrature rule with respect to binning if the "model" u is a known test function. The second test case, which is discussed in Section 5.4.2, consists of the DLC 1.2 load case, where only the wind conditions are considered uncertain (and the sea state is fixed at nominal conditions). The full five-dimensional load case is then assessed purely by means of a quadrature rule and contains the five aforementioned parameters, modeling both the wind and the sea state. The results of this case (which follows the IEC standard [85, 86] to a large extent) are discussed in Section 5.4.3.

5.4.1. Test functions

The five-dimensional Genz test functions [66] are reconsidered, which are six functions $u_1, \ldots, u_6$ constructed specifically to assess the accuracy of integration methods (see Table 4.1 on page 89, reproduced as Table 5.2 in this chapter for sake of completeness). The goal is to calculate $\mathbb{E}[u]$ defined by measurement data (see (5.2)), i.e. the goal is to use binning and an implicit quadrature rule to estimate

$$\mathbb{E}[u_g] = \frac{1}{K} \sum_{k=0}^{K-1} u_g(\widehat{\mathbf{y}}_k), \text{ for } g = 1, \ldots, 6.$$

The data under consideration is formed by the measurements from the IJmuiden meteorological mast (see Section 5.2.2), scaled to the domain of definition of the Genz test functions. Therefore, $\widehat{\mathbf{y}}_k$ denotes $\mathbf{y}_k$ after scaling of all data to $\Omega = [0,1]^d$ (with $d = 5$). As the functions u_g are known exactly, no seeds are employed in this test case.

5

Binning (see Section 5.3.1) is performed by fixing a number $B = 1, \ldots, 7$, constructing B bins in each direction (obtaining B^5 bins), and removing all bins with zero weight. Then the implicit quadrature rule (see Section 5.3.2) is constructed such that its number of nodes is equal to the number of obtained bins. The vectors $\mathbf{a}$ and $\mathbf{b}$, that define the Genz test function under consideration, are selected randomly in the unit hypercube and $\mathbf{a}$ is scaled such that $\|\mathbf{a}\|_2 = 5/2$. The experiment is repeated 100 times (i.e. with 100 random vectors $\mathbf{a}$ and $\mathbf{b}$) to obtain representative values of the integration error:

$$\overline{e}_N = \frac{1}{100} \sum_{k=1}^{100} e_N^{(k)}, \text{ with } e_N^{(k)} = \left| \mathbb{E}[u_g^{(k)}] - \mathcal{A}_N u_g^{(k)} \right|.$$

Here, $u_g^{(k)}$ is the g-th Genz test function constructed with the k-th pair of random vectors $\mathbf{a}$ and $\mathbf{b}$ and N denotes the number of bins with non-zero weight or, equivalently, the number of nodes of the quadrature rule. The obtained results are depicted in Figure 5.5.

First, notice that the quadrature rule consistently outperforms binning. The convergence behavior of binning is approximately equal for all functions, as binning does not leverage smoothness of the function due to its local approximation. Moreover a large number of bins is necessary to obtain an accurate estimation. On the other hand, the convergence behavior of the quadrature rule clearly reflects the attributes of the test functions (see Table 5.2). Similar behavior was observed in Chapters 3 and 4, with the key difference that here no artificially generated data is being used, but the data encompasses measurements done at the North Sea (see Section 5.2.2). The functions u_1, u_2, u_3, and u_4 are smooth and can be approximated accurately using a polynomial of low degree. The integration error of these functions decays fastest. The fifth and sixth Genz test functions are not smooth and it is not straightforward to approximate these functions by means of a polynomial. This can also be observed in the convergence behavior of the quadrature rule, as it clearly converges slower than the other test functions.

5.4.2. Two-dimensional verification load case

The two-dimensional case under consideration is a case in which the wind speed and wind direction vary, but the sea state is considered to be fixed. The distribution of the wind speed and wind direction are as described in Section 5.2.1 and 5.2.2, so the wind speed follows the data and the wind direction is uniformly distributed between −12° and 12°. The parameters describing the sea state are fixed at their mean values obtained from the data, which are $H_s = 1.46\,\text{m}$, $T_p = 6.76\,\text{s}$, and $\theta_{\text{wind/wave}} = -2.11°$.

In this case, the conventional approach is to bin the wind speeds using bins of 2 m/s and to bin the wind direction into three bins of 8°. The wind speed varies between (approximately) 0 m/s and 30 m/s, yielding 15 bins in this dimension (these are also depicted in Figure 5.3a), resulting in a total number of 45 bins. The frequency of occurrence of each bin is determined using the number of measurements in the bin, which yields a non-zero weight for any bin. For a fair comparison, the implicit quadrature rule is constructed such that it contains 45 nodes based on the same two parameters.

For both cases, five seeds per bin (or node) are used and by means of rainflow cycle counting the weighted equivalent loads are determined. For the quadrature rule,

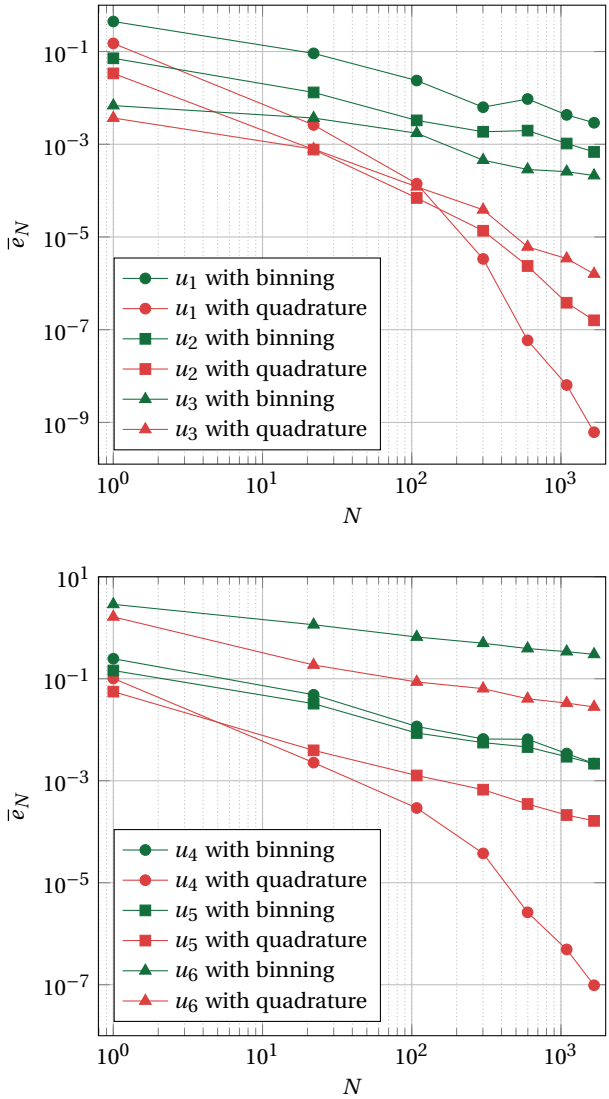


Figure 5.5: Mean integration error of the five-dimensional Genz test functions for various numbers of nodes, determined using binning (with the same number of bins in each dimension) or a quadrature rule.

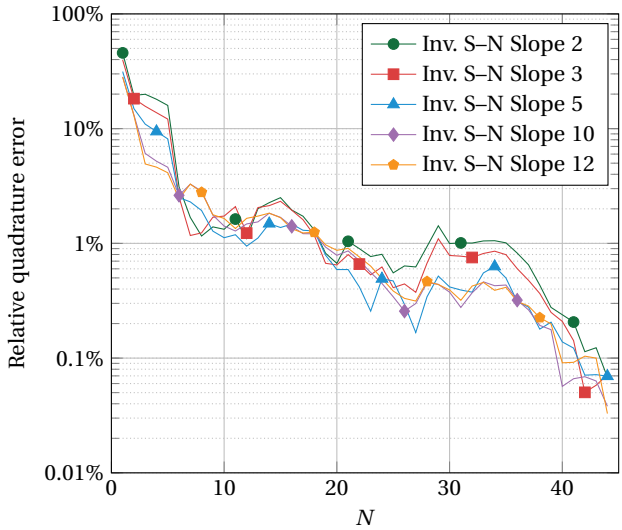
Table 5.3: Equivalent loads of the two-dimensional load case, determined using various inverse S–N slopes, acting on various components of the turbine, calculated with three different methods.

Component	Inv. S–N slope	Binning (five seeds)	Quadrature rule (five seeds)	Quadrature rule (balanced seeds)
Rotating hub	2	$6.37 \cdot 10^5$ N	$6.37 \cdot 10^5$ N	$6.32 \cdot 10^5$ N
(In longitudinal direction)	3	$4.65 \cdot 10^5$ N	$4.62 \cdot 10^5$ N	$4.59 \cdot 10^5$ N
	5	$4.39 \cdot 10^5$ N	$4.39 \cdot 10^5$ N	$4.35 \cdot 10^5$ N
	10	$4.92 \cdot 10^5$ N	$5.02 \cdot 10^5$ N	$4.97 \cdot 10^5$ N
	12	$5.12 \cdot 10^5$ N	$5.23 \cdot 10^5$ N	$5.19 \cdot 10^5$ N
Blade root	2	$1.85 \cdot 10^7$ Nm	$1.85 \cdot 10^7$ Nm	$1.85 \cdot 10^7$ Nm
(Flap-wise moment)	3	$1.34 \cdot 10^7$ Nm	$1.34 \cdot 10^7$ Nm	$1.34 \cdot 10^7$ Nm
	5	$1.16 \cdot 10^7$ Nm	$1.15 \cdot 10^7$ Nm	$1.15 \cdot 10^7$ Nm
	10	$1.20 \cdot 10^7$ Nm	$1.21 \cdot 10^7$ Nm	$1.19 \cdot 10^7$ Nm
	12	$1.25 \cdot 10^7$ Nm	$1.26 \cdot 10^7$ Nm	$1.23 \cdot 10^7$ Nm
Yaw bearing	2	$8.03 \cdot 10^5$ N	$8.01 \cdot 10^5$ N	$7.93 \cdot 10^5$ N
(In longitudinal direction)	3	$6.00 \cdot 10^5$ N	$5.93 \cdot 10^5$ N	$5.87 \cdot 10^5$ N
	5	$5.63 \cdot 10^5$ N	$5.57 \cdot 10^5$ N	$5.49 \cdot 10^5$ N
	10	$6.41 \cdot 10^5$ N	$6.37 \cdot 10^5$ N	$6.22 \cdot 10^5$ N
	12	$6.76 \cdot 10^5$ N	$6.70 \cdot 10^5$ N	$6.53 \cdot 10^5$ N

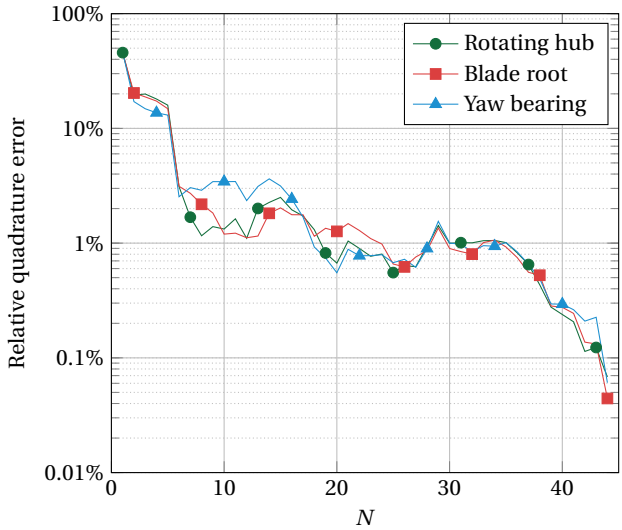
the seeds are balanced using the procedure from Section 5.3.3. Seed balancing is not applied to binning, as we want to keep binning as close to the conventional approach as possible. The number of required `BLADED` runs of the balanced seeds in conjunction with a quadrature rule decreases from 225 to 173, which is (approximately) 77 % of the original cost. Some obtained loads for several components calculated using these three approaches (i.e. binning and quadrature rule with five seeds, and a quadrature rule with balanced seeds) are tabulated in Table 5.3.

The difference between the loads determined with a quadrature rule (both determined with five seeds or balanced seeds) and those determined with binning is less than 5%. Strictly speaking, it is not evident whether the quadrature rule or binning yields a more accurate answer in this case, as no exact values of the expected values of the loads are known. However, based on the results of Section 5.4.1, we have a strong indication that the quadrature rules yield more accurate estimates. The largest variation is in the loads on the yaw bearing, which is likely due to its large sensitivity to the varying environmental conditions. The contrary is true for the blade root, which is significantly less sensitive to environmental variations. This will also be observed in the next section, where the five-dimensional case is considered.

One main advantage of the quadrature rule approach is that its accuracy can be assessed without any additional costly aeroelastic calculations using the removal procedure discussed in Section 5.3.2 (and originally introduced in Chapter 3). To this end, equivalent loads have been determined using $N = 1, \ldots, 44$ nodes, and these are compared with the value from Table 5.3. As discussed in Section 5.3.2 there are multiple



(a) Rotating hub



(b) Inv. S–N Slope 2

Figure 5.6: Relative quadrature error for various S–N slopes (top) or various components (bottom) considering the two-dimensional load case without seed balancing.

sequences of 45 nested quadrature rules. The errors reported here are averaged over five sequences of nested quadrature rules, i.e.

$$\overline{e}_N = \frac{1}{5}\sum_{k=1}^{5} \frac{\left|L_{N,k}^{\mathrm{eq}} - L_{45}^{\mathrm{eq}}\right|}{L_{45}^{\mathrm{eq}}}.$$

Here, $L_{N,k}^{\mathrm{eq}}$ refers to calculating the equivalent load using N nodes of the k-th quadrature rule sequence. The error is scaled with the loads from Table 5.3, such that loads of different order of magnitude and with different characteristics can be compared.

The obtained quadrature errors are depicted in Figure 5.6. Notice that all reported errors decay approximately three orders of magnitude, to a point where the error is in the order of 0.1 % of the equivalent load. The errors decay rapidly for small N and the rate of decay reduces for increasing N. This is likely due to the fact that the error decays algebraically (as u_5 did in Section 5.4.1), i.e. its decay is approximately a straight line if depicted on a log-log plot. However, significantly more `BLADED` runs are necessary to confirm this claim numerically (e.g. see the number of nodes considered in the construction of Figure 5.5). The top figure depicts the convergence for the equivalent loads of the rotating hub for various slopes of the S–N curve. It demonstrates that the behavior of the error is similar for different values of the S–N slope, as all lines decay rapidly. The bottom figure demonstrates that the behavior of the error is more or less independent of the specific component under consideration.

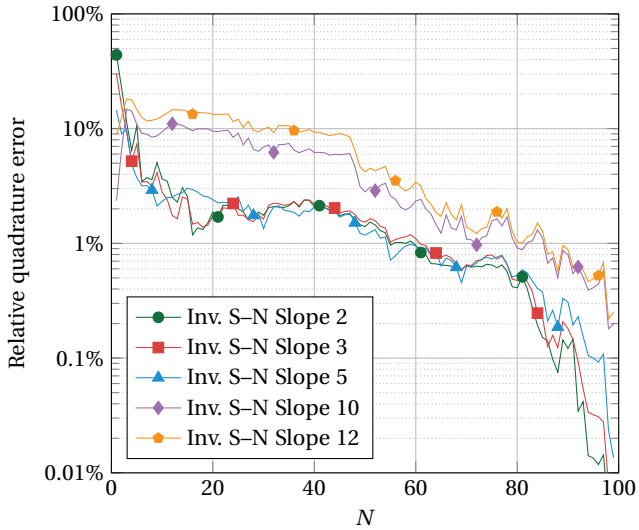
5.4.3. Five-dimensional Design Load Case

The five-dimensional load case consists of DLC 1.2 as described in the IEC standard [85, 86]. The random parameters are in this case the five aforementioned parameters, consisting of wind speed, wind direction, wind/wave misalignment, significant wave height, and peak spectral period.

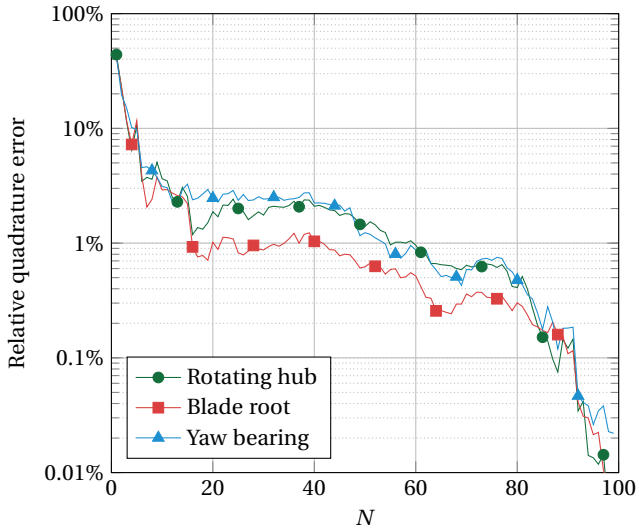
Applying binning to this case is infeasible, as the number of bins with non-zero frequency of occurrence equals 6 308, which would result in approximately 30 000 evaluations of the aeroelastic simulation code. Even after optimizing the number of seeds using seed balancing from Section 5.3.3 the number of evaluations is still 23 219, which remains intractable.

Instead, an implicit quadrature rule is determined consisting of 100 nodes. This number of nodes is larger than the number of nodes from the previous section, to account for the higher dimensionality of the problem. Furthermore, the number of nodes is sufficient to obtain a small error (which can be assessed during the simulation using the methods discussed in Section 5.3.2). The seeds are optimized using the method from Section 5.3.3, resulting in 419 individual aeroelastic simulation code evaluations. This is a major reduction compared to the number of bins, since the number of `BLADED` evaluations is less than 2% of that of binning. The equivalent loads acting on various components are summarized in Table 5.4.

The loads of the rotating hub and blade root are for the wind turbine and offshore site under consideration relatively close to the values that were calculated in the two-dimensional load case. These loads are apparently less sensitive to environmental variations which could be due to their high position on the turbine. The loads acting



(a) Rotating hub



(b) Inv. S–N Slope 2

Figure 5.7: Relative quadrature error for various S–N slopes (top) or various components (bottom) considering the five-dimensional load case with seed-balancing.

Table 5.4: Equivalent loads of the five-dimensional load case, determined using various inverse S–N slopes, acting on various components of the turbine, using the full five-dimensional DLC 1.2 fatigue load case calculated with the seed-balanced quadrature rule.

Inv. S–N slope	Rotating hub (In longitudinal direction)	Blade root (Flap-wise moment)	Yaw bearing (In longitudinal direction)
2	$6.36 \cdot 10^5$ N	$1.84 \cdot 10^7$ Nm	$8.06 \cdot 10^5$ N
3	$4.66 \cdot 10^5$ N	$1.34 \cdot 10^7$ Nm	$6.11 \cdot 10^5$ N
5	$4.48 \cdot 10^5$ N	$1.15 \cdot 10^7$ Nm	$7.65 \cdot 10^5$ N
10	$5.50 \cdot 10^5$ N	$1.20 \cdot 10^7$ Nm	$1.59 \cdot 10^6$ N
12	$6.05 \cdot 10^5$ N	$1.24 \cdot 10^7$ Nm	$1.82 \cdot 10^6$ N

on the yaw bearing are significantly larger if a random sea is considered. As the yaw bearing is the component that directly links the tower (which is placed directly in the sea) and the rotating hub (which is connected to the blades), its loads are sensitive to variations in both the wind and the sea. Contrary to the two-dimensional case, the sea state is random, so these variations are much larger.

To assess the accuracy of the obtained loads, the convergence is assessed in a similar way as done in the previous section: by removal of nodes from the quadrature rule with 100 nodes, sequences of quadrature rules are constructed that are used to assess the accuracy. There is a minor subtlety in this case: the number of seeds is optimized, so the nodes of the smaller quadrature rules are evaluated using an incorrect number of seeds. We do not correct for this, so the estimated error might be slightly larger. The results are gathered in Figure 5.7.

Compared to the two-dimensional test case, the errors of the quadrature rules are slightly larger, which is either due to the larger dimensionality of the problem or due to the unbalanced seeds used to construct these figures. Nonetheless, the errors clearly decay to a relative error smaller than 1%. For larger slopes of the S–N curve, the error is larger, which is likely due to the higher power used in the expression that is integrated (this increases the constant of proportionality discussed in Section 5.3.3). Again, the convergence behavior is possibly algebraic, though significantly more simulations are necessary to fully quantify the error. This is out of the scope of this chapter.

Concluding, this test case demonstrates that the proposed quadrature rule is capable of accurately approximating weighted equivalent loads in standardized form using less than 500 `BLADED` evaluations, which is a significant reduction compared to the approximately 30 000 `BLADED` evaluations that binning requires.

5.5. Conclusion

A novel approach based on quadrature rules has been proposed to calculate wind turbine fatigue loads as described by Design Load Case 1.2, which is standardized by IEC [85, 86]. The quadrature rule under consideration is the implicit quadrature rule as derived in Chapter 4, with the key properties that it has positive weights and can be constructed

using measurement data. It is based on polynomial approximation, which leverages smoothness in the model to achieve high accuracy. To demonstrate the efficiency of the new approach, it has been compared to binning, the conventional approach. In both approaches, the number of seeds per bin or node can be balanced to maximize accuracy in the available computational time. The environmental parameters are based on real offshore measurements in the North Sea and the wind turbine under consideration is the NREL 5MW reference wind turbine.

Both quadrature and binning approaches have been applied to a simplified two-dimensional load case, for which it has been demonstrated that the accuracy of the quadrature rule is comparable to that of binning. Moreover, the error of the quadrature rule converges algebraically upon addition of nodes.

The main advantages of the quadrature rule, i.e. a significant reduction of computational time with similar accuracy, have been demonstrated by considering the full load case, which is governed by five random parameters. In this case, binning is infeasible. The accuracy of the quadrature rule has again been assessed numerically, confirming that the error of the rule decays rapidly.

Throughout this chapter, all results are generated based on five seeds per node. A possible improvement is to vary the number of seeds per node, for which a further study of the significance of the seed error compared to the quadrature error needs to be performed.

Overall, the results demonstrate that for fatigue load cases our proposed quadrature rule forms a highly promising alternative to binning with significantly lower computational cost and similar accuracy. Therefore, a topic for future research is to extend the numerical integration framework to other load scenarios, e.g. fatigue load cases with more uncertain parameters (where the benefit of using a quadrature rule is even larger) or ultimate load cases (where no rainflow cycle counting is applied).

6

Bayesian model calibration with interpolating polynomials

The quadrature rules discussed in the previous chapters are efficient approaches to assess the effect of uncertainty arising in external sources on the output of the model. In the derivation of these approaches it is assumed that the model and the description of the probability distribution of its parameters (i.e. by means of a probability density function or by means of samples) do not contain error. This is not always the case in practice, since models and parameter values contain inherent assumptions or biases that could yield overconfident or meaningless predictions if not properly accounted for.

A common approach to systematically estimate the unknown model parameters of non-linear models such that the model uncertainty is captured to a certain extent is Bayesian model calibration. The main result of such an approach is the *posterior*, which is a distribution over the model parameters. It can be used to infer predictions that incorporate the uncertainty of the model. However, the approach is computationally expensive by design, as it requires a large number of model runs to obtain the statistics on the model parameters. In this chapter, an algorithm is proposed to replace the costly model with a computationally cheap surrogate. The key component of the algorithm is formed by weighted Leja nodes, which are used to adaptively ensure that the surrogate can be used to determine an accurate posterior.

6.1. Introduction

Systematically estimating model parameters from measurements is a problem of frequent occurrence in many fields of engineering and many different approaches exist to solve this problem. The approach followed in this thesis is of a stochastic nature: the unknown parameters are modeled using probability distributions and possible values

This chapter is based on the following article: L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes. *Communications in Computational Physics*, **27**(1):33–69, 2020. DOI: 10.4208/cicp.oa-2018-0218. arXiv: 1802.02035 [math.NA].

of these parameters are inferred using Bayesian statistics. This is often called *Bayesian model calibration*, and has been introduced briefly in Section 2.2. The mathematical foundation of these techniques is relatively straightforward, but naively assessing the obtained statistics requires many model evaluations. The latter challenge is the key focus of this chapter.

Bayesian model calibration [95, 153, 158] is a systematic way to calibrate the parameters of a computational model. By means of a statistical model that describes the relation between the model and the data, the calibrated parameters are modeled using a probability density function (PDF), called the *posterior*, which is obtained using Bayes' law. This density function can consequently be used to assess the uncertainty in the model and to infer predictions. The focus of this chapter is mainly on calibration of the model, whereas the focus of Chapter 7 shifts to Bayesian prediction. The calibration approach has already been applied many times, for example to calibrate the closure parameters of turbulence models [34, 52]. A similar example is considered in this chapter.

Possibly the largest drawback of Bayesian model calibration is the expensive sampling procedure that is necessary. The posterior depends to a large extent on the model, which is only known implicitly (e.g. a computer code numerically solving a partial differential equation), so drawing a sample from the posterior is usually done using generally applicable Markov chain Monte Carlo (MCMC) methods [80, 122], which require many expensive model evaluations, as outlined in Section 2.2.2 on page 24. Improvements have been made to accelerate these MCMC methods, e.g. the DREAM algorithm [186] or adaptive sampling [193]. Replacing the sampling procedure itself is also possible, e.g. methods based on sparse grids [33, 117] or Approximate Bayesian Computation [6, 40, 111]. However, this encompasses stringent assumptions on the statistical model or still requires many model runs as the shape of the posterior is unknown.

A different approach is followed in the current chapter. In essence the approach of Marzouk et al. [120] is followed, which has been used several times in literature [1, 11, 121, 138, 149, 204, 205, 206]. The key idea is to replace the model in the calibration step with a *surrogate* (or *response surface*) that approximates the computationally expensive model. MCMC can then be used to sample the obtained posterior without a large computational overhead.

Various approaches to construct this surrogate in a Bayesian context exist, for example Gaussian process emulators [173] or non-intrusive polynomial approximations [201] as discussed in Section 2.1.2. The focus of this thesis is on the latter, because polynomial approximations provide high order (up to exponential) convergence for sufficiently smooth functions. Contrary to the commonly used pseudo-spectral projection methods, which are known as generalized Polynomial Chaos Expansions, we choose to use interpolation of the computationally expensive model. The reason for this is that the error of a polynomial interpolant is usually measured using the absolute error (the L^∞_ρ-norm), contrary to the mean squared error (the L^2_ρ-norm) that is used for the pseudo-spectral approaches. As the model is used as input in the Bayesian analysis, having absolute error bounds on the surrogate significantly simplifies the analysis. Moreover, the convergence of a pseudo-spectral expansion deteriorates significantly if the surrogate is not constructed using the statistical model [112]. This happens in particular if the expansion

is constructed with respect to the prior (which is the usual approach) and the likelihood is very informative (i.e. the relative entropy between the prior and the likelihood is high).

The interpolating polynomial is constructed using weighted Leja nodes [90, 130]. Ideally, Leja nodes weighted using the posterior are used, but these are generally unavailable since the posterior is computationally costly. Therefore in this chapter weighted Leja nodes are extended in a novel way to *adaptively* refine the interpolating polynomial using an approximation of the posterior. As extensive theory about interpolation polynomials exists (e.g. [84]), we can prove convergence of the estimated posterior with mild assumptions on the likelihood, provided that widely used conventional Leja nodes yield a converging polynomial interpolant. This extends previous work [11, 120], in which the likelihood is assumed to be Gaussian. The end result is an interpolating polynomial that can be used in conjunction with the likelihood and the prior to obtain statistics of the posterior.

To demonstrate the applicability of our methodology, we will employ three different classes of test problems. The first class consist of functions that are known explicitly and can be evaluated fast and accurately. We will use these to show the effectiveness of our nodal set compared to commonly used methods. The second class consists of problems that are defined implicitly, but do not require significant computational power to solve. For this, we employ the one-dimensional Burgers' equation. In this case, it is possible to compare the estimated posterior with a posterior determined using Monte Carlo methods. The last class consists of problems of such large complexity that a quantitative comparison with a true posterior is not possible anymore. As example we consider the calibration of closure coefficients of the Spalart–Allmaras turbulence model.

This chapter is set up as follows. First, we discuss Bayesian model calibration and introduce the adaptively weighted Leja nodes. In Section 6.3 the theoretical properties of the algorithm are studied and its convergence is assessed. Section 6.4 contains numerical tests that show evidence of the theoretical findings and in Section 6.5 conclusions are drawn.

6.2. Bayesian model calibration with a surrogate

The focus is on the stochastic calibration of computationally expensive (possibly implicitly defined) models. As done throughout this thesis, this model is denoted by $u\colon \Omega \to \mathbb{R}$, with $\Omega \subset \mathbb{R}^d$ $(d = 1, 2, \dots)$. Without loss of generality, we assume that u is a scalar quantity and that u depends on d parameters, which we will denote as usual using a vector $\mathbf{x} = (x_1, \dots, x_d)^{\mathrm{T}} \in \Omega$. Contrary to previous chapters, where these parameters were based on physical quantities, in this chapter one can think of $\mathbf{x}$ as parameters inherent to the model, such as fitting parameters or other closure parameters.

The goal of Bayesian model calibration is to infer statistical quantities of the model parameters, given observations of the process modeled by u. To this end, we assume that a vector of observations $\mathbf{z} = (z_1, \dots, z_n)^{\mathrm{T}}$ is given, with $z_k \in \mathbb{R}$. This vector can be provided by various means, for example by measurements or by the results of a high-fidelity model. Using parameters $\mathbf{x}$, a statistical model is formulated describing a relation between the model $u(\mathbf{x})$ and the data $\mathbf{z}$ by means of random variables that model among others discrepancy, error, and uncertainty. For example, these random variables account

for measurement errors and numerical tolerances. Using Bayesian statistics [65], the posterior of the parameters is formulated by means of a PDF.

The nomenclature and notation from Chapter 2 is reconsidered here. Hence let $q(\mathbf{x})$ be the prior, a PDF that models the prior knowledge of $\mathbf{x}$ obtained for example through physical constraints, assumptions, or previous experiments. The likelihood $q(\mathbf{z}\,|\,\mathbf{x})$ is obtained through the statistical relation between the model u and the data $\mathbf{z}$. Possibly the most straightforward example is $z_k = u(\mathbf{x}) + \varepsilon_k$, where $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^{\mathrm{T}}$ is assumed to be multivariate Gaussian distributed with mean $\mathbf{0}$ and covariance matrix Σ. As illustrated in Chapter 2, this yields the following likelihood:

$$q(\mathbf{z}\,|\,\mathbf{x}) \propto \exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}}\Sigma^{-1}\mathbf{d}\right], \text{ with } \mathbf{d} \text{ a vector such that } d_k = z_k - u(\mathbf{x}). \tag{6.1}$$

The vector $\mathbf{d}$ is the so-called misfit. Bayes' law is applied to obtain the posterior $q(\mathbf{x}\,|\,\mathbf{z})$, i.e.

$$q(\mathbf{x}\,|\,\mathbf{z}) \propto q(\mathbf{z}\,|\,\mathbf{x})\, q(\mathbf{x}). \tag{6.2}$$

The posterior can be used to infer statistical quantities of the parameters of the model, e.g. by determining the expectation or the maximum a posteriori (MAP) estimate. The uncertainty of these parameters can be quantified by determining the moments of the posterior.

Note that the posterior depends on the likelihood, which requires an expensive evaluation of the model (see (6.1)). Therefore sampling the posterior using MCMC methods [80, 122] is typically intractable for the computationally expensive models considered in this thesis.

Vector-valued models u can be incorporated in this framework straightforwardly, although the likelihood requires minor modifications. Typically an observation operator is introduced that restricts u to the locations where measurement data is available. We will discuss an example of this in Section 6.4.3.

Throughout this chapter we assume that the likelihood is a continuously differentiable, Lipschitz continuous function of the misfit $\mathbf{d}$ or (more generally speaking) of the model u. This is true for the multivariate Gaussian likelihood and (more generally speaking) for any likelihood which has additive errors (see [95] for more examples in the context of Bayesian model calibration). There are no further constraints on the structure of the likelihood and the prior in this chapter, but we do not accommodate the calibration of *hyperparameters*, i.e. parameters of the distributions introduced in the statistical model (an example would be the calibration of the standard deviation of $\boldsymbol{\varepsilon}$). Moreover, we assume the prior is not improper, i.e. it is a well-defined distribution with

$$\int_{\Omega} q(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1.$$

Even though this prohibits the usage of a uniform prior on an unbounded interval, in practice our methods can be applied in such a setting.

The outline of the proposed calibration procedure is as follows. Let u_N be an interpolating surrogate of u computed using $N+1$ nodes and concomitant model evaluations. Using u_N, an estimated posterior can be determined, which is used to obtain the

$(N+2)$-nd node. The steps are repeated until convergence is observed. Finally MCMC can be applied to the obtained posterior, because the computationally expensive model is replaced with an explicitly known surrogate.

First, we briefly reintroduce the interpolation polynomial for sake of completeness. Then the nodal set we will use, the Leja nodes, will be introduced.

6.2.1. Interpolation methods

Polynomial interpolation as introduced in Section 2.1.2 is used in this chapter to construct a surrogate of the computationally expensive model. To this end, let $u\colon \Omega \to \mathbb{R}$ with $\Omega \subset \mathbb{R}^d$ be a continuous function. Consider a sequence of monomials $\varphi_0, \dots, \varphi_N$, sorted graded reverse lexicographically in this chapter, and the function space these polynomials span, denoted as before by $\Phi_N = \operatorname{span}\{\varphi_0, \dots, \varphi_N\}$. Then using a nodal set $X_N = \{\mathbf{x}_0, \dots, \mathbf{x}_N\}$ and evaluations of u at each node (i.e. $u(\mathbf{x}_k)$ for $k = 0, \dots, N$) the goal is to determine a polynomial $u_N \in \Phi_N$ such that

$$u_N(\mathbf{x}_k) = u(\mathbf{x}_k), \text{ for } k = 0, \dots, N.$$

Univariate interpolation

In the case of $d = 1$, it is well known that if all nodes are distinct the interpolation polynomial can be stated explicitly using Lagrange interpolating polynomials, as discussed in Section 2.1.2 and defined by (2.5). In particular, the following expression is obtained:

$$u_N(x) = (\mathcal{L}_N u)(x) := \sum_{k=0}^{N} \ell_k^N(x) u(x_k), \text{ with } \ell_k^N(x) = \prod_{\substack{j=0 \\ j \neq k}}^{N} \frac{x - x_j}{x_k - x_j}. \tag{6.3}$$

Here $\mathcal{L}_N$ is a linear operator that yields a polynomial of degree N, which we will denote as u_N as above. Recall that by construction the Lagrange basis polynomials ℓ_k^N have the property $\ell_k^N(x_j) = \delta_{k,j}$ (i.e. $\ell_k^N(x_j) = 1$ if $j = k$ and $\ell_k^N(x_j) = 0$ otherwise). Therefore $u_N(x_k) = u(x_k)$ for all k, such that it is indeed an interpolating polynomial.

The barycentric notation [7] can be used to numerically evaluate the interpolating polynomial given a nodal set (which is unconditionally stable [82]). It follows by rewriting (6.3) in the following form:

$$u_N(x) = \left(\sum_{k=0}^{N} \frac{v_k}{x - x_k} u(x_k) \right) \Bigg/ \left(\sum_{k=0}^{N} \frac{v_k}{x - x_k} \right), \text{ with } v_k = 1 \Bigg/ \prod_{\substack{j=0 \\ j \neq k}}^{N} (x_k - x_j)\,. \tag{6.4}$$

Common factors in the so-called barycentric weights v_k cancel. Therefore this formulation can be used to accurately evaluate the interpolating polynomial, provided that the nodal set has asymptotically constant barycentric weights (such as Chebyshev nodes [7]), i.e. $(\max_k |v_k|)/(\min_k |v_k|) = \mathcal{O}(1)$.

Multivariate interpolation

The Lagrange interpolating polynomials can also be formulated explicitly in a multivariate setting, though arguably less intuitively, by defining them in terms of the determinant of a Vandermonde matrix:

$$u_N(\mathbf{x}) = (\mathcal{L}_N u)(\mathbf{x}) := \sum_{k=0}^{N} \ell_k^N(\mathbf{x}) u(\mathbf{x}_k), \text{ with } \ell_k^N(\mathbf{x}) = \frac{\det V(\mathbf{x}_0,\dots,\mathbf{x}_{k-1},\mathbf{x},\mathbf{x}_{k+1},\dots,\mathbf{x}_N)}{\det V(\mathbf{x}_0,\dots,\mathbf{x}_{k-1},\mathbf{x}_k,\mathbf{x}_{k+1},\dots,\mathbf{x}_N)}, \tag{6.5}$$

where $V(\mathbf{x}_0,\dots,\mathbf{x}_N)$ is the $(N+1)\times(N+1)$ Vandermonde matrix with respect to the nodal set $\{\mathbf{x}_0,\dots,\mathbf{x}_N\}$, i.e. as introduced before (e.g. (4.4) on page 71):

$$V(\mathbf{x}_0,\dots,\mathbf{x}_N) = \begin{pmatrix} \varphi_0(\mathbf{x}_0) & \cdots & \varphi_0(\mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ \varphi_N(\mathbf{x}_0) & \cdots & \varphi_N(\mathbf{x}_N) \end{pmatrix}.$$

Multivariate interpolation by means of this Vandermonde matrix is only well defined if this matrix is non-singular, in which case X_N is called a poised interpolation sequence. All nodal sequences constructed in this chapter are (by construction) poised.

There exist various other monomial orders, for example for the purpose to construct a sparse grid [134]. Also adaptive choices have been studied [130]. Often these approaches leverage structure in the underlying distribution by decomposing it in d univariate distributions. Such efficient approaches cannot be applied to the context of this chapter, because it is rarely the case that the posterior can be decomposed in d univariate distributions, due to the asymmetry in the model and the measurement data. Nonetheless, the framework and algorithms proposed in this chapter can easily accommodate different monomial orders.

Evaluating a multivariate interpolating polynomial numerically can be done in various ways. A commonly used approach is to choose $\varphi_j(\mathbf{x})$ to be orthogonal polynomials (e.g. Chebyshev or Legendre polynomials) and to compute the interpolating polynomial by solving the linear system from (2.4) (see page 12) for coefficients a_k. This constitutes computing u_N by solving

$$u_N(\mathbf{x}_j) = \sum_{k=0}^{N} a_k \varphi_k(\mathbf{x}_j) = u(\mathbf{x}_j), \text{ for } j = 0,\dots,N. \tag{6.6}$$

Orthogonal polynomials form a linear combination of monomials, so mathematically speaking the Lagrange interpolating polynomial or the linear system from (6.6) yield exactly the same result.

The nodal sets used in this chapter are defined using the determinant of the Vandermonde matrix. Therefore a QR factorization [73] of the Vandermonde matrix is used to solve (6.6). Consequently the QR factorization is reused to determine the nodal set (see Section 6.2.2 for details).

6.2.2. Weighted Leja nodes

In Chapter 4 three important properties of quadrature rules have been discussed: accuracy, positivity of the weights, and nesting. A nodal set used for interpolation ideally

has similar traits. More specifically, for the purpose of Bayesian model calibration we desire an algorithm to determine a nodal set X_N for any N such that it has the following properties:

1. **Accuracy:** the nodal sets should yield an accurate posterior. We are mainly interested in estimating the posterior, i.e. it is strictly speaking not necessary to have an accurate surrogate model.

2. **Nested:** we require $X_i \subset X_j$ for $i < j$, such that the obtained interpolant can be refined by reusing existing model evaluations.

3. **Weighting:** the goal is to determine the next node based on the posterior obtained so far. Therefore the nodal set should support weighting to incorporate arbitrary probability distributions.

In this chapter we consider weighted Leja nodes, which form a sequence of nodes and are therefore by definition nested. We proceed by defining univariate Leja nodes and generalizing these to multivariate Leja nodes.

The definition of weighted Leja nodes is by induction. Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be a bounded PDF and let $\{x_0, \dots, x_N\}$ be a sequence of Leja nodes. Then the next node is defined as follows:

$$x_{N+1} := \underset{x\in\mathbb{R}}{\arg\max}\, \rho(x)|\det V(x_0, \dots, x_N, x)| = \underset{x\in\mathbb{R}}{\arg\max}\, \rho(x)|x - x_0||x - x_1|\cdots|x - x_N|. \quad (6.7)$$

This maximization problem does not necessarily have a unique solution. To ensure that a solution exists and polynomial approximation is well defined, it is necessary to assume that the polynomials are dense in the space of continuous functions equipped with the ∞-norm weighted with ρ, i.e. the following norm:

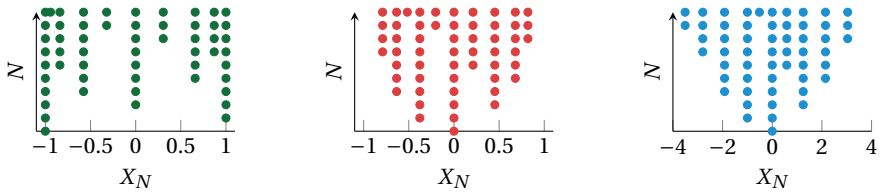
$$\|f\|_\rho = \|f\rho\|_\infty = \sup_{x\in\mathbb{R}} |f(x)\rho(x)|.$$

This norm is simply called the ρ-norm in this chapter. It is important to emphasize that this ρ-norm is conceptually different from the L^∞_ρ-norm used in Section 2.1.2. If there are multiple values maximizing (6.7), we pick the one with smallest x to ensure that the sequence is reproducible. The initial node x_0 is defined as the smallest global maximum of $\rho(x)$.

The requirement that the polynomials must be dense in the space of continuous functions equipped with the ρ-norm is non-trivial. It is among others the case if Ω is bounded or if $\rho(x) \propto \exp(-|x|^\alpha)$ with $\alpha \geq 1$ [130], which constitute all cases discussed in this chapter. It is necessary that ρ has finite moments, but this is not sufficient. An elaborate discussion is out of the scope of this chapter and the interested reader is referred to Lubinsky [113].

Notice that definition (6.7) can be rewritten as follows:

$$x_{N+1} = \underset{\rho(x)>0}{\arg\max}\left(\log\rho(x) + \sum_{k=0}^{N} \log|x - x_k|\right). \quad (6.8)$$



(a) Uniform ("unweighted") **(b)** Beta ($\alpha = \beta = 4$) **(c)** Standard normal

Figure 6.1: Univariate Leja sequences for various numbers of nodes and various well-known distributions.

If ρ is bounded from below and above, i.e. $A \leq \rho(x) \leq B$ for $0 < A < B$ and $\rho(x)$ has bounded support, the sum $\log|x - x_0| + \cdots + \log|x - x_N|$ will dominate the maximal value for large N. Hence for any x the value of the sum will increase (but remains bounded as $\rho(x)$ has bounded support) and $\rho(x)$ will remain constant (as ρ is independent of N). This implies that for ρ that are bounded from below and above, the influence of the weighting function decreases as N increases.

Unweighted Leja nodes are defined with the uniform weighting function on $[-1, 1]$. We want to emphasize that multiplying the weighting function with a constant yields an identical sequence. This property is very useful for our purposes, as it allows us to neglect the constant of proportionality (often called the evidence) in Bayes' law (see (6.2)).

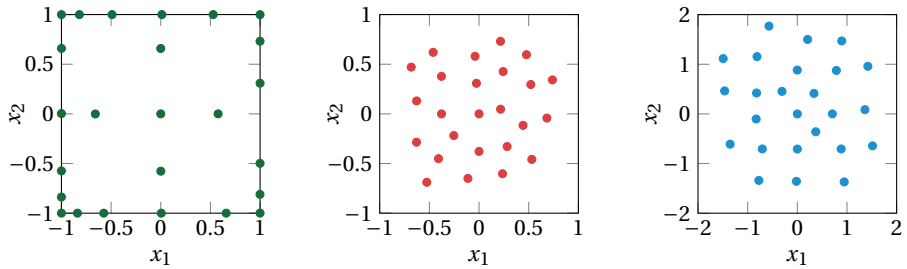
Examples of these sequences are depicted in Figure 6.1. Throughout this chapter, univariate Leja nodes are determined by applying Newton's method to the derivative of the logarithm of the maximization problem above, i.e. (6.8) is solved instead of (6.7). By determining all local maxima between two consecutive nodes in parallel, large numbers of nodes can be calculated fast and accurately (as the maximization function is smooth between two nodes). Numerical cancellation is kept minimal by using extended precision arithmetic (with machine epsilon approximately 10^{-19}).

The definition of univariate Leja nodes from (6.7) can be generalized straightforwardly to a multidimensional setting in a similar way as we did in Section 6.2.1 with interpolation. To this end, let $\rho\colon \mathbb{R}^d \to [0, \infty)$ be a multivariate PDF. Let $\mathbf{x}_0 \in \mathbb{R}^d$ be an initial node with $\rho(\mathbf{x}_0) > 0$, which is selected as the minimal value that maximizes ρ (if there are multiple, select the smallest considering a lexicographical ordering). Then given the nodes $\mathbf{x}_0, \ldots, \mathbf{x}_N$, the next node $\mathbf{x}_{N+1}$ is defined as follows:

$$\mathbf{x}_{N+1} := \underset{\mathbf{x} \in \mathbb{R}^d}{\arg\max}\, \rho(\mathbf{x}) \left|\det V(\mathbf{x}_0, \ldots, \mathbf{x}_N, \mathbf{x})\right|. \tag{6.9}$$

Here, V is the Vandermonde matrix defined in Section 6.2.1. The absolute value of the determinant of V is independent of the set of polynomials that is used to construct V, so the definition is mathematically the same for both monomials and orthogonal polynomials.

Determining multivariate Leja nodes is less trivial compared to univariate nodes and is typically done by randomly (or quasi-randomly) sampling the space of interest and selecting the node that results in the highest determinant. It is significantly more



(a) Uniform ("unweighted") **(b)** Multivariate Beta ($\alpha = \beta = 4$) **(c)** Multivariate normal

Figure 6.2: Multivariate Leja sequences of 25 nodes using various well-known distributions.

complicated to reliably apply Newton's method in this case, as the space cannot be easily partioned in regions where the local maxima reside. To reach a comparable accuracy, it is important to be able to use a large number of samples, so it must be possible to calculate the determinant fast. We suggest to calculate the determinant by an extended QR factorization of the matrix consisting of the first $N+1$ rows of $V(\mathbf{x}_0,\ldots,\mathbf{x}_N,\mathbf{x})$. Then for a different random sample, the $(N+2)$-nd column containing $\mathbf{x}$ can be updated efficiently by applying a rank-1 update. If a QR factorization has been calculated to determine the interpolating polynomial (see Section 6.2.1), it can be reused here. As a rank-1 update is an efficient procedure, a large number of samples can be used and therefore we assume that the approximation error is negligible in this case. Examples of Leja sequences defined by (6.9) can be found in Figure 6.2.

6.2.3. Calibration using Leja nodes

In this section a weighting function is derived which is used in the procedure discussed in the previous section, with the goal to approximate the posterior. Theoretical details are provided in Section 6.3. First the rationale behind the weighting function is discussed. Then the weighting function is introduced formally and the mathematical derivation it is based upon is presented. Finally, the single free parameter of the weighting function is discussed.

Rationale

If the posterior is known explicitly and samples can be readily drawn from it, it is possible to determine weighted Leja nodes with weighting function $\rho(\mathbf{x}) = q(\mathbf{x}\,|\,\mathbf{z})$. Such an interpolant is specifically tailored to computing integrals with respect to the posterior (this is commonly known as Bayesian prediction and is considered in more detail in Chapter 7). However, the posterior is generally not explicitly available because it depends on the model u, which in itself is not known explicitly and can only be determined on (finitely many) nodes. Therefore the need arises for an interpolation sequence that approximates u such that the posterior determined with this approximation is accurate.

To this end, let $q_N(\mathbf{x}\,|\,\mathbf{z})$ be the posterior determined using $u_N(\mathbf{x})$, i.e. the interpolant

of u using $N+1$ nodes. If the likelihood is according to (6.1), q_N is as follows:

$$q_N(\mathbf{x}\,|\,\mathbf{z}) \propto q(\mathbf{x}) \exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}}\Sigma^{-1}\mathbf{d}\right], \text{ with } \mathbf{d} \text{ a vector such that } d_k = z_k - u_N(\mathbf{x}).$$

We will use the definition of the weighted Leja nodes from (6.9) to determine the next node. The natural idea is to construct $q_{N+1}(\mathbf{x}\,|\,\mathbf{z})$ (i.e. a new approximation of the posterior) by determining a new weighted Leja node using $q_N(\mathbf{x}\,|\,\mathbf{z})$ (i.e. the existing approximation of the posterior). Such a sequence can be numerically unstable, because it solely places nodes in regions where the approximate posterior is high and therefore yields spurious oscillations in other regions in the domain.

The key idea is to balance the accuracy of the interpolant with the accuracy of the posterior. There are various methods to do this, but we choose to temper the effect of the (possibly inaccurate) approximate posterior by adding a constant value ζ to it. The higher this ζ, the more the posterior tends to the prior. In Section 6.3.2 it is demonstrated that for any $\zeta > 0$, the interpolant constructed with these weighted Leja nodes has (at least) the same asymptotic convergence rate as an interpolant determined with weighted Leja nodes without adaptivity. If ζ is chosen correctly, the approximate posterior is already accurate for moderately small N.

The adaptive weighting function

To introduce this construction formally, we assume that a function $\mathcal{Q}\colon \mathbb{R} \to [0,\infty)$ exists such that

$$q(\mathbf{z}\,|\,\mathbf{x}) = \mathcal{Q}(u(\mathbf{x})), \tag{6.10}$$

where $\mathcal{Q}$ is typically a PDF which follows from the statistical model. In the example discussed in (6.1) $\mathcal{Q}$ is a Gaussian PDF, i.e.

$$\mathcal{Q}\colon \mathbb{R} \to [0,\infty), \text{ with } \mathcal{Q}(u) \propto \exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}}\Sigma^{-1}\mathbf{d}\right] \text{ and } d_k = z_k - u. \tag{6.11}$$

We assume that the function $\mathcal{Q}$ is globally Lipschitz continuous and continuously differentiable. Many statistical models used in a statistical setting yield Lipschitz continuous $\mathcal{Q}$, because a bounded continuously differentiable function $\mathcal{Q}(u)$ with $\mathcal{Q}'(u) \to 0$ for $u \to \pm\infty$ is Lipschitz continuous. The domain of definition of $\mathcal{Q}$ is the image of the model u, so functions $\mathcal{Q}$ that are only Lipschitz continuous in the set described by the image of u also fit in this framework (for example the Gamma distribution on the positive real axis).

The weighting function proposed in this chapter, called $q_N^{(\zeta)}$, is chosen such that it balances between $q_N(\mathbf{x}\,|\,\mathbf{z})$ and $q(\mathbf{x})$. It follows from the mean value theorem, which will be discussed in more detail later. It explicitly depends on N and a free parameter ζ, and is defined as follows:

$$q_N^{(\zeta)}(\mathbf{x}\,|\,\mathbf{z}) := |\mathcal{Q}'(u_N(\mathbf{x}))|\,q(\mathbf{x}) + \zeta\, q(\mathbf{x}), \text{ where } \zeta > 0 \text{ is a free parameter.} \tag{6.12}$$

So, if $\mathbf{x}_0, \ldots, \mathbf{x}_N$ are the first $N+1$ Leja nodes, $\mathbf{x}_{N+1}$ is determined as follows:

$$\mathbf{x}_{N+1} = \arg\max_{\mathbf{x}} q_N^{(\zeta)}(\mathbf{x}\,|\,\mathbf{z})\,|\det V(\mathbf{x}_0, \ldots, \mathbf{x}_N, \mathbf{x})|. \tag{6.13}$$

Here the derivative $\mathcal{Q}'$ is with respect to u, i.e.

$$\mathcal{Q}'(u(\mathbf{x})) = \frac{\partial \mathcal{Q}}{\partial u}(u(\mathbf{x})).$$

We want to emphasize that for the evaluation of $\mathcal{Q}'(u_N(\mathbf{x}))$ no costly evaluation of the full model u is necessary, since $\mathcal{Q}'$ is an explicit expression. In the example from (6.1), $\mathcal{Q}'$ is as follows:

$$\mathcal{Q}' : \mathbb{R} \to \mathbb{R}, \text{ with } \mathcal{Q}'(u) \propto -\frac{1}{2}\left(\mathbf{1}^{\mathrm{T}}\Sigma^{-1}\mathbf{d} + \mathbf{d}^{\mathrm{T}}\Sigma^{-1}\mathbf{1}\right)\exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}}\Sigma^{-1}\mathbf{d}\right] \text{ and } d_k = z_k - u,$$

with $\mathbf{1} = (1, 1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^n$.

Mean value theorem

The weighting function q_N as defined in (6.12) follows naturally by applying the mean value theorem to the error of the approximate posterior. This introduces the derivative $\mathcal{Q}'$ in the expression. To this end, let a fixed $\mathbf{x}$ be given, and apply the mean value theorem as follows:

$$\begin{aligned} |q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z})| &= |\mathcal{Q}(u_N(\mathbf{x})) - \mathcal{Q}(u(\mathbf{x}))| q(\mathbf{x}) \\ &= |\mathcal{Q}'(\xi)||u_N(\mathbf{x}) - u(\mathbf{x})| q(\mathbf{x}) \\ &= |\mathcal{Q}'(u_N(\mathbf{x})) + \zeta_{\mathbf{x}}||u_N(\mathbf{x}) - u(\mathbf{x})| q(\mathbf{x}), \end{aligned} \tag{6.14}$$

with ξ an (unknown) value between $u_N(\mathbf{x})$ and $u(\mathbf{x})$ and $\zeta_{\mathbf{x}} = \mathcal{Q}'(\xi) - \mathcal{Q}'(u_N(\mathbf{x}))$. Essentially $\zeta_{\mathbf{x}}$ is used to represent higher order derivatives of $\mathcal{Q}$ in this expression. The value of $\zeta_{\mathbf{x}}$ depends on $\mathbf{x}$ and on the model u, which is not explicitly known. By further expanding $\mathcal{Q}'$, it can be shown that $\zeta_{\mathbf{x}}$ scales with $|u_N(\mathbf{x}) - u(\mathbf{x})|$, provided that $\mathcal{Q}$ is twice differentiable with bounded second order derivative:

$$\begin{aligned} \zeta_{\mathbf{x}} &= \mathcal{Q}'(\xi) - \mathcal{Q}'(u_N(\mathbf{x})) \\ &= \frac{1}{2}\mathcal{Q}''(\widehat{\xi})(u_N(\mathbf{x}) - u(\mathbf{x})), \text{ for a } \widehat{\xi} \text{ between } u_N(\mathbf{x}) \text{ and } u(\mathbf{x}). \end{aligned}$$

Hence if $u_N(\mathbf{x}) \to u(\mathbf{x})$ for $N \to \infty$ and if $\mathcal{Q}''$ is bounded (or if the divided difference of $\mathcal{Q}'$ is bounded), it holds that $\zeta_{\mathbf{x}} \to 0$ for $N \to \infty$. In this chapter, the constant $\zeta_{\mathbf{x}}$ is used to measure how far the likelihood of the interpolant is from the likelihood of the true model. The idea is to add a Leja node $\mathbf{x}_{N+1}$ where the error in the posterior is large, though such that the interpolant remains stable. The weighting function $q_N^{(\zeta)}$ as introduced before follows by taking the ∞-norm in $\mathbf{x}$ on both sides of (6.14):

$$\begin{aligned} \| q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z}) \|_\infty &= \| \mathcal{Q}(u_N(\mathbf{x})) - \mathcal{Q}(u(\mathbf{x})) \|_\infty \\ &= \| |\mathcal{Q}'(\xi)| (u_N(\mathbf{x}) - u(\mathbf{x})) q(\mathbf{x}) \|_\infty \\ &\leq \| \left(|\mathcal{Q}'(u_N(\mathbf{x}))| + \zeta\right)\left(u_N(\mathbf{x}) - u(\mathbf{x})\right) q(\mathbf{x}) \|_\infty \\ &= \| (u_N(\mathbf{x}) - u(\mathbf{x})) q_N^{(\zeta)}(\mathbf{x}) \|_\infty, \end{aligned}$$

with $\zeta \geq |\zeta_{\mathbf{x}}| = |\mathcal{Q}'(\xi) - \mathcal{Q}'(u_N(\mathbf{x}))|$ for all $\mathbf{x}$.

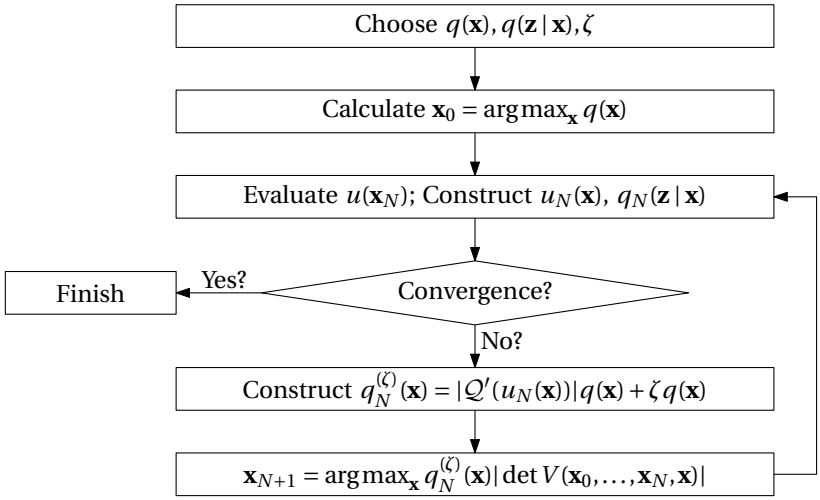


Figure 6.3: Schematic overview of the algorithm considered in this chapter.

The algorithm proposed in this chapter is to (iteratively) firstly determine $q_N^{(\zeta)}$, secondly determine $\mathbf{x}_{N+1}$ using (6.13), and finally determine $u(\mathbf{x}_{N+1})$ and reconstruct the interpolant (which yields u_{N+1} and consequently $q_{N+1}^{(\zeta)}(\mathbf{x}\,|\,\mathbf{z})$). This algorithm is sketched in Figure 6.3. Convergence can be assessed in various ways, for example using the ∞-norm or the Kullback–Leibler divergence. We will mainly focus on the ∞-norm, as determining the Kullback–Leibler divergence in high-dimensional spaces is numerically challenging.

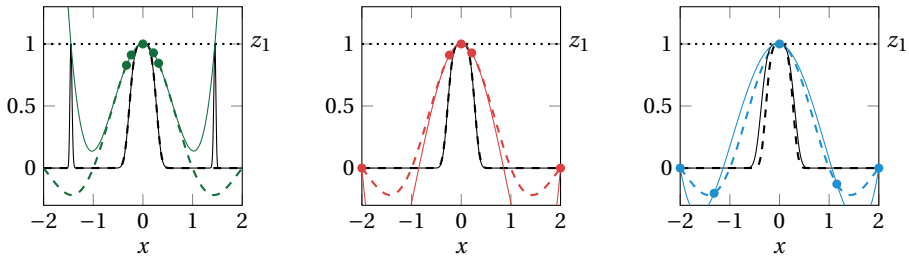
The exact value of $\zeta_{\mathbf{x}}$ is not known a priori and depends on $\mathbf{x}$. Nonetheless, we will demonstrate that for any $\zeta > 0$ it holds that $\|u - u_N\|_\infty \to 0$ (for $N \to \infty$), provided that "conventional" weighted Leja nodes produce a converging interpolant. If $u_N \to u$ for $N \to \infty$, the exact value of ζ converges to 0, hence any value of ζ will work for sufficiently large N. We will further study the convergence of this method in Section 6.3.

Choice of ζ

To illustrate the behavior of the weighting function as defined by (6.12), examples of interpolants obtained using Leja nodes weighted using $q_N^{(\zeta)}$ in conjunction with the exact posterior are depicted in Figure 6.4. Here the parameter x of the univariate function $u(x) = \mathrm{sinc}(x) = \sin(x)/x$ is "calibrated" using the Gaussian likelihood from (6.11) with $\sigma = 1/10$, a uniform prior defined on $[-2,2]$, and a single data point at $z_1 = 1$. Hence the exact posterior is as follows:

$$q(x\,|\,z_1) \propto \begin{cases} \exp\left[-\frac{1}{2\sigma^2}|u(x) - z_1|^2\right] & \text{if } |x| \le 2, \\ 0 & \text{otherwise.} \end{cases}$$

The weighting function under consideration is $q_N^{(\zeta)}(x) = |\mathcal{Q}'(u(x))| + \zeta$, where u is used instead of u_N to illustrate the effect of ζ.



(a) $\zeta = 0$ **(b)** $\zeta = 1/100$ **(c)** $\zeta = 100$

Figure 6.4: Interpolation of the sinc function using 5 weighted Leja nodes with respect to the posterior using a tempering parameter ζ. The model $u(x)$ and (unscaled) posterior $q(x \mid z_1)$ are depicted in color and in black respectively. The solid line represents the result constructed by means of interpolation. The "true" model and posterior are depicted using a dashed line.

If $\zeta = 0$ (no tempering) the interpolant is indeed accurate with respect to the posterior (i.e. the weighted $q(x \mid z_1)$-norm), but yields an incorrect approximate posterior because the interpolant intersects the value of the data incorrectly around $x = \pm 1.5$. These spurious oscillations disappear for larger N, but for different test cases this is not necessarily the case (as it requires global analyticity). For $\zeta = 100$, it is guaranteed that $\zeta \geq |\mathcal{Q}'(\xi) - \mathcal{Q}'(u_N(x))|$ for all x, but the nodes determined with that value are, due to the large variations in the determinant of the Vandermonde matrix, not sensitive to small variations in the approximate posterior, and are therefore pointwise close to unweighted Leja nodes (e.g. compare Figure 6.4c with Figure 6.1a). The best strategy is to take a small non-zero value of ζ, which balances posterior accuracy with stability. For such a small non-zero value, the second and third node are basically unweighted Leja nodes (and end up on the boundary). This does demonstrate the importance of tempering on the effect of the approximate posterior, which becomes especially important if the function u is not globally analytic (but "only" continuous).

The key point in obtaining a converging interpolant is that $\zeta > 0$. If $\zeta = 0$, the inaccuracy of u_N can significantly deteriorate the convergence (see Figure 6.4), except possibly if u is globally analytic. If the goal is to optimize ζ, we suggest an adaptive approach based on a heuristic. Start with $\zeta = \zeta_0 > 0$ and for each iteration, multiply ζ with a constant $\alpha > 1$ if the error in the posterior increases and divide ζ by α if the interpolation error decreases. The error can be estimated by comparing two consecutive approximate posteriors. This procedure is however not necessary to obtain convergence for the examples in this chapter, for which a fixed value of ζ is sufficient.

6.3. Convergence of the posterior

In this section the convergence of the estimated posterior to the true posterior is studied, denoted as follows:

$$\| q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z}) \|_\infty \to 0, \text{ for } N \to \infty.$$

It is difficult to theoretically demonstrate this, since the convergence rate of interpolants constructed with Leja nodes is only known in some specific cases. However, we will demonstrate that the convergence rate of an interpolant determined with adaptively weighted Leja nodes is similar to the rate of an interpolant determined with Leja nodes without adaptivity, such that all results on the convergence of these conventional Leja nodes carry over.

The analysis is split into two parts. First, in Sections 6.3.1 and 6.3.2 the focus is on the model, i.e. it is assessed in which cases $\|u_N - u\|_{q(\mathbf{x})} \to 0$ for $N \to \infty$ (where $q(\mathbf{x})$ denotes the prior). In Section 6.3.1 convergence properties of interpolation methods are briefly reviewed. A more extensive discussion can be found in Section 2.1.2 (see page 11). In Section 6.3.2 the focus is specifically on Leja nodes, a case that will be mostly assessed numerically. Moreover, the close relation between adaptively weighted Leja nodes and Leja nodes without adaptivity is considered.

The second part of the analysis consists of demonstrating that the posterior converges if the interpolant converges. Specifically, in Section 6.3.3 the following is demonstrated:

$$\|q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z})\|_\infty \leq L\|u_N - u\|_{q(\mathbf{x})}, \text{ with } L \text{ a constant independent of } N.$$

The conventional way of describing the distance between two distributions is by means of the Kullback–Leibler divergence. In Section 6.3.4 it is proved that if the interpolant converges to the true model, the Kullback–Leibler divergence between the approximate posterior and the true posterior converges to zero. Moreover, the rate of convergence doubles.

6.3.1. Accuracy of interpolation methods

The accuracy of interpolation methods can be assessed in two ways: using *pointwise* error bounds which are typically based on Taylor expansions and *global* error bounds which are typically based on the Lebesgue inequality, as previously introduced in Chapter 2. In this chapter, the focus is primarily on the latter type.

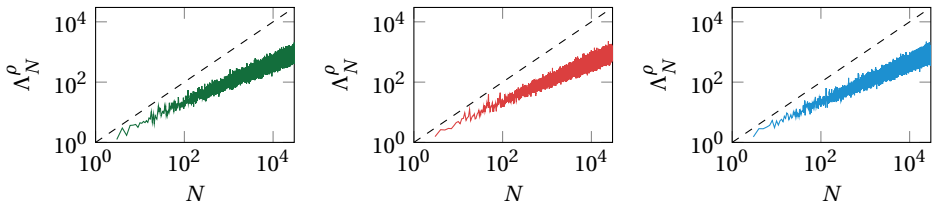
Let $\varphi_0, \ldots, \varphi_N$ be $N+1$ monomials sorted graded reverse lexicographically and consider $\Phi_N = \text{span}\{\varphi_0, \ldots, \varphi_N\}$. Let $u \in C(\Omega)$ be a continuous function representing the computational model with $\|u\|_\infty < \infty$ if not stated otherwise. It is well known that $C(\Omega)$ equipped with the norm $\|\cdot\|_\infty$ forms a Banach space, provided that Ω is compact.

The previously discussed Lebesgue inequality follows readily. Let $X_N = \{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$ be a set of interpolation nodes and let $\mathcal{L}_N \colon C(\Omega) \to \Phi_N$ be the Lagrange interpolation operator from (6.3) that determines the interpolating polynomial given the nodal set X_N. Then as introduced in Section 2.1.2 (see (2.9) on page 14), the Lebesgue inequality reads as follows [84]:

$$\|\mathcal{L}_N u - u\|_\infty \leq (1 + \Lambda_N) \inf_{\varphi \in \Phi_N} \|\varphi - u\|_\infty. \tag{6.15}$$

with

$$\Lambda_N := \|\mathcal{L}_N\|_\infty = \sup_{\mathbf{x} \in \Omega} \sum_{k=0}^{N} |\ell_k^N(\mathbf{x})|.$$



(a) Uniform **(b)** Standard Gaussian **(c)** Beta ($\alpha = \beta = 4$)

Figure 6.5: The weighted Lebesgue constant of weighted Leja nodes for three different distributions. The dashed line depicts $\Lambda_N = N$.

The Lebesgue constant Λ_N is the operator norm of $\mathcal{L}_N$ induced by the norm $\|\cdot\|_\infty$ discussed above. The multivariate Lagrange basis polynomials $\ell_k^N(\mathbf{x})$ are as in (6.5).

In the procedure used for calibration, nodes are determined using a weighting function ρ. To assess the accuracy of nodes that use weighting, we reconsider the ρ-norm $\|u\|_\rho = \|\rho\, u\|_\infty$. Here, we assume that $\rho\colon \Omega \to \mathbb{R}$ is a bounded PDF, such that $C(\Omega)$ equipped with the norm $\|\cdot\|_\rho$ forms a Banach space. The space Ω is allowed to be unbounded, in contrast to the unweighted case. The unweighted case is a special case of the weighted case.

Using this norm we can derive a similar estimate as (6.15) by introducing [90]:

$$\Lambda_N^\rho := \|\mathcal{L}_N\|_\rho = \sup_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} \frac{\rho(\mathbf{x})}{\rho(\mathbf{x}_k)} |\ell_k^N(\mathbf{x})|.$$

Here, Λ_N^ρ is called the weighted Lebesgue constant, i.e. the norm of the operator $u \mapsto \rho\mathcal{L}_N(u/\rho)$. We call the result the weighted Lebesgue inequality:

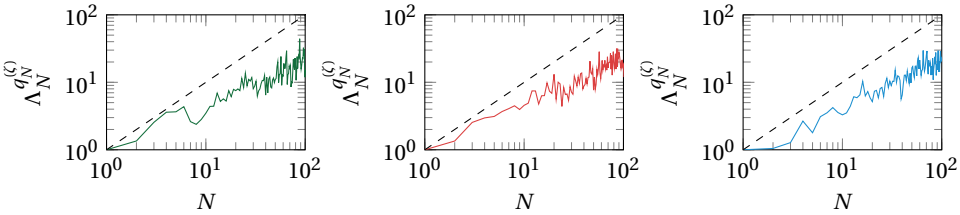
$$\|\mathcal{L}_N u - u\|_\rho \leq (1 + \Lambda_N^\rho) \inf_{\varphi\in\Phi_N} \|\varphi - u\|_\rho.$$

The Lebesgue inequality does not readily provide means to estimate the order of convergence, as discussed extensively previously in this thesis. Jackson's inequality [87, 140] can be used for this purpose, but this inequality is in principle only applicable to the usual ∞-norm. Jackson's inequality has been extended to the weighted case, see Lubinsky [113] and the references therein, though the technical details are rather involved. Important for this chapter is that if u is Lipschitz continuous (or continuous and bounded on a compact domain) then a sublinearly growing Lebesgue constant provides a converging interpolant.

6.3.2. Lebesgue constant of Leja nodes

It is both an advantage and a disadvantage that the Lebesgue constant solely depends on the nodal set: we do not have to take the model into account to estimate the accuracy, but the resulting estimate does not leverage any properties of the model.

There exist many nodal sets with a logarithmically growing Lebesgue constant, which is asymptotically the optimal growth. For example, Chebyshev nodes (i.e. the nodes from



(a) $\zeta = 0$ **(b)** $\zeta = 1/100$ **(c)** $\zeta = 100$

Figure 6.6: The weighted Lebesgue constant of adaptively weighted Leja nodes using a Gaussian likelihood with $\sigma = 1/10$, a uniform prior on $[-2,2]$, a single data point $z_1 = 1$, and the function $u(x) = \mathrm{sinc}(x)$. The dashed line depicts $\Lambda_N = N$.

the Clenshaw–Curtis quadrature rule) have $\Lambda_N = \mathcal{O}(\log N)$ [84]. Moreover, we already stated that the Chebyshev nodes are nested such that the nodes for $N = 2^l + 1$ (for integer l) are contained in the nodes for $N = 2^{l+1} + 1$. However, the Chebyshev nodes are only defined in an *unweighted* setting. Another well-known example is formed by equidistant nodes, which have an exponentially growing Lebesgue constant. This can be observed by interpolating Runge's function.

Although it is known that the Lebesgue constant of both weighted and unweighted univariate Leja sequences grows sub-exponentially [90, 177, 178], the exact growth (or a strict upper bound) is not known. We have therefore numerically determined the Lebesgue constant for N up to 30 000 for several distributions and observed $\Lambda_N^{\rho} \leq N$, see Figure 6.5. For the purpose of these figures, the Leja nodes and the Lebesgue constant have been determined by applying Newton's method to the derivative of the function, as described in Section 6.2.2.

Except for some specific cases (such as the unit disk [32]), not much is known about the Lebesgue constant in the multivariate case. Unfortunately, as far as the authors know, it is difficult to determine the Lebesgue constant of multivariate Leja nodes accurately enough to create a similar plot as Figure 6.5. For small numbers of nodes ($N \leq 100$) and low dimensionality ($d \leq 5$) the Lebesgue constants seem to grow similarly, but the sampling procedure significantly deteriorates the accuracy. Moreover this number is too small to draw conclusions about general asymptotic behavior. Nonetheless, the numerically determined growth of the Lebesgue constant is sufficient for the purposes in the current chapter (since we have $N \leq 100$ throughout this chapter). Contrary to the Lebesgue constant, large numbers of multivariate Leja nodes can be determined efficiently by means of sampling, as evaluating the maximization function is significantly more straightforward (see Section 6.2.2).

These results do not carry over straightforwardly to the case of adaptively weighted Leja nodes where the weighting function depends on the number of nodes. However, for reasonably small N the Lebesgue constant can be assessed numerically. To this end, we have determined the Lebesgue constant $\Lambda_N^{q_N^{(\zeta)}}$, i.e. the Lebesgue constant weighted with $q_N^{(\zeta)}$ from (6.12), of adaptively weighted Leja nodes using the aforementioned example

of a Gaussian likelihood in conjunction with the function $u(x) = \operatorname{sinc}(x)$ (see Figure 6.6). Determining Leja nodes in this case is still relatively straightforward, but determining the Lebesgue constant accurately is significantly less trivial due to the varying weight function, so we limit ourselves to at most 100 nodes. Even though the weighting function now depends on the number of nodes, it appears that the Lebesgue constant still grows sublinearly. This result, in conjunction with the numerical results from Section 6.4, is a strong indication that weighted Leja nodes as proposed in this chapter indeed yield an interpolant that can be used to construct an accurate approximate posterior. Notice that slow growth of $\Lambda_N^{q_N^{(\zeta)}}$ implies slow growth of Λ_N^q and vice versa (where q denotes the prior), which can be seen as follows:

$$\begin{aligned}
\Lambda_N^q &= \sup_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} \frac{q(\mathbf{x})}{q(\mathbf{x}_k)} |\ell_k^N(\mathbf{x})| \leq \sup_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} \frac{\|\mathcal{Q}'\|_\infty + \zeta}{\zeta} \frac{q_N^{(\zeta)}(\mathbf{x})}{q_N^{(\zeta)}(\mathbf{x}_k)} |\ell_k^N(\mathbf{x})| \leq \left(1 + \frac{\|\mathcal{Q}'\|_\infty}{\zeta}\right) \Lambda_N^{q_N^{(\zeta)}}, \\
\Lambda_N^{q_N^{(\zeta)}} &= \sup_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} \frac{q_N^{(\zeta)}(\mathbf{x})}{q_N^{(\zeta)}(\mathbf{x}_k)} |\ell_k^N(\mathbf{x})| \leq \sup_{\mathbf{x}\in\Omega} \sum_{k=0}^{N} \frac{\|\mathcal{Q}'\|_\infty + \zeta}{\zeta} \frac{q(\mathbf{x})}{q(\mathbf{x}_k)} |\ell_k^N(\mathbf{x})| \leq \left(1 + \frac{\|\mathcal{Q}'\|_\infty}{\zeta}\right) \Lambda_N^q.
\end{aligned} \tag{6.16}$$

Furthermore, this expression can be used to demonstrate that results about the growth of the Lebesgue constant to a certain extent carry over to the setting of adaptively determined nodes. To see this, notice that if $\mathbf{x} \in \Omega$ is given and $\mathbf{x}_0, \ldots, \mathbf{x}_N$ are adaptively weighted Leja nodes, it holds for all $k = 0, \ldots, N$ that

$$\begin{aligned}
\zeta q(\mathbf{x}) |\det V(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{x})| &\leq q_k^{(\zeta)}(\mathbf{x}) |\det V(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{x})| \\
&\leq (\zeta + \|\mathcal{Q}'\|_\infty) q(\mathbf{x}_k) |\det V(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_k)|.
\end{aligned}$$

Hence let $\widehat{q}(\mathbf{x})$ be as follows:

$$\widehat{q}(\mathbf{x}) = \begin{cases} \zeta + \|\mathcal{Q}'\|_\infty & \text{if } \mathbf{x} = \mathbf{x}_k \text{ for any } k = 0, \ldots, N, \\ \zeta & \text{otherwise.} \end{cases}$$

Then it holds for all $k = 0, \ldots, N$ that

$$\widehat{q}(\mathbf{x}) |\det V(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{x})| \leq \widehat{q}(\mathbf{x}_k) |\det V(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_k)|.$$

Hence there exists a single weighting function $\widehat{q}$ that defines these nodes. Moreover, following the same derivation as (6.16), it holds that

$$\Lambda_N^q \leq \left(1 + \frac{\|\mathcal{Q}'\|_\infty}{\zeta}\right) \Lambda_N^{\widehat{q}} \text{ and } \Lambda_N^{\widehat{q}} \leq \left(1 + \frac{\|\mathcal{Q}'\|_\infty}{\zeta}\right) \Lambda_N^q, \tag{6.17}$$

where $\Lambda_N^{\widehat{q}}$ is used instead of $\Lambda_N^{q_N^{(\zeta)}}$ (i.e. the weighting function is independent of N).

Concluding, the Lebesgue constant of adaptively weighted Leja nodes grows asymptotically at least as slow as the Lebesgue constant of Leja nodes weighted with $\widehat{q}$. Furthermore, the growth of the Lebesgue constant weighted with the prior is similar to the growth of the Lebesgue constant weighted with $\widehat{q}$. As discussed before, it is difficult to assess these bounds theoretically, though the Lebesgue constant can often be assessed numerically. Notice that it is essential that $\zeta > 0$ for this result to hold, since otherwise the constant in (6.17) can become unbounded.

6.3.3. Convergence of the posterior

In this section the convergence of the method is studied. In particular, the interest is in the error of the estimated posterior in the ∞-norm, given convergence of the interpolant. It is demonstrated that, provided that the interpolant converges, a posterior constructed with the interpolant converges.

As discussed previously, let $q(\mathbf{x})$, $q(\mathbf{z} \mid \mathbf{x})$, and $q(\mathbf{x} \mid \mathbf{z})$ be the prior, likelihood, and posterior respectively and let u be the model as usual. We assume an interpolant u_N is given such that $\|u_N - u\|_{q(\mathbf{x})} \to 0$ for $N \to \infty$. Such an interpolant can for example be constructed with adaptively weighted Leja nodes, as discussed extensively in the previous section, but this is not explicitly assumed here (e.g. Leja nodes weighted with the prior also suit). Let $q_N(\mathbf{z} \mid \mathbf{x})$ and $q_N(\mathbf{x} \mid \mathbf{z})$ be the likelihood and the posterior respectively, both constructed with this interpolant (i.e. $q_N(\mathbf{z} \mid \mathbf{x}) = \mathcal{Q}(u_N(\mathbf{x}))$). The main result, stated in Theorem 6.1 below, is that if the interpolant converges to the model, the approximate posterior converges to the posterior. We do not need differentiability of $\mathcal{Q}$ in this general case, but require $\mathcal{Q}$ to be Lipschitz continuous.

Theorem 6.1. *Let $u\colon \Omega \to \mathbb{R}$ be a continuous function and let u_N be the interpolant of u constructed with N nodes. Suppose*

$$\|u_N - u\|_{q(\mathbf{x})} \leq C Q_N,$$

with $Q_N \to 0$ for $N \to \infty$, and C a positive constant (independent of N). Assume the likelihood (i.e. the function $\mathcal{Q}$) is Lipschitz continuous.

Then

$$\|q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z})\|_\infty \leq K Q_N,$$

where K is a positive constant.

Proof. Recall the definition of $\mathcal{Q}$ from (6.10): $q(\mathbf{z} \mid \mathbf{x}) = \mathcal{Q}(u(\mathbf{x}))$. Let L be the Lipschitz constant of $\mathcal{Q}$. Convergence readily follows:

$$\begin{aligned}
\|q_N(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{x} \mid \mathbf{z})\|_\infty &= \|(q_N(\mathbf{z} \mid \mathbf{x}) - q(\mathbf{z} \mid \mathbf{x}))q(\mathbf{x})\|_\infty \\
&= \|[\mathcal{Q}(u_N(\mathbf{x})) - \mathcal{Q}(u(\mathbf{x}))]q(\mathbf{x})\|_\infty \\
&\leq L\|(u_N(\mathbf{x}) - u(\mathbf{x}))q(\mathbf{x})\|_\infty \\
&= L\|u_N(\mathbf{x}) - u(\mathbf{x})\|_{q(\mathbf{x})} \\
&\leq LCQ_N.
\end{aligned}$$

■

If u_N converges to u in the $q(\mathbf{x})$-norm, the estimated posterior converges to the true posterior with at least the same rate of convergence, e.g. exponential if $Q_N \propto A^{-N}$ (for $A > 1$) and algebraic if $Q_N \propto N^{-\alpha}$ for $\alpha > 0$. This concludes the proof of Theorem 6.1, extending previous work [11, 30, 121, 203] to the interpolating framework of this chapter.

6.3.4. The Kullback–Leibler divergence

The Kullback–Leibler divergence is often used in a Bayesian setting to measure distance between distributions. It is defined as follows, given two probability density functions $p(x)$ and $q(x)$ defined on a set Ω:

$$D_{\mathrm{KL}}\big(p(x) \parallel q(x)\big) = \int_\Omega p(x) \log \frac{p(x)}{q(x)} \,\mathrm{d}x.$$

The Kullback–Leibler divergence is always positive, vanishes if (and only if) $p \equiv q$, and is finite if $p(x) = 0$ implies $q(x) = 0$ (here, it is used that $\lim_{x\to 0} x \log x = 0$). The interest is in proving bounds on $D_{\mathrm{KL}}\big(q_N(\mathbf{x}\,|\,\mathbf{z}) \,\|\, q(\mathbf{x}\,|\,\mathbf{z})\big)$ or $D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \,\|\, q_N(\mathbf{x}\,|\,\mathbf{z})\big)$. This is non-trivial due to the logarithm in the integral. The analysis consists of two parts: first it is studied in which cases the Kullback–Leibler divergence converges to zero and then the convergence rate is derived.

Convergence of the Kullback–Leibler divergence

To prove convergence, first pointwise convergence of the logarithm is proved, then this is extended to convergence of the integral using Fatou's lemma, and finally it is concluded that the Kullback–Leibler divergence converges. As the Kullback–Leibler divergence is defined for probability density functions, we have to incorporate the scaling of the posterior. To this end, let γ_N and γ be defined as follows:

$$\begin{aligned} \gamma &= \int_\Omega q(\mathbf{z}\,|\,\mathbf{x})\, q(\mathbf{x})\, \mathrm{d}\mathbf{x}, \\ \gamma_N &= \int_\Omega q_N(\mathbf{z}\,|\,\mathbf{x})\, q(\mathbf{x})\, \mathrm{d}\mathbf{x}. \end{aligned} \tag{6.18}$$

To start off, the following lemma provides pointwise convergence of $\log \frac{q_N(x)}{q(x)}$, i.e. $\log \frac{q_N(x)}{q(x)}$ converges for each $x \in \Omega$. We omit the proof.

Lemma 6.2. *Let $g_n \colon \Omega \to \mathbb{R}$ be a series of functions with $g_n(x) \to g(x)$ for $n \to \infty$, for all $x \in \Omega$. Assume $g_n > 0$ for all n and $g > 0$. Then*

$$\log \frac{g_n(x)}{g(x)} \to 0, \text{ for } n \to \infty, \text{ for all } x \in \Omega.$$

Note that the generalization to *uniform* convergence is non-trivial, since that would require that the convergence of $\log \frac{q_N(x)}{q(x)}$ is independent of x, which is not obviously the case. However, by definition the Kullback–Leibler divergence does not require uniform convergence, but only convergence in the integral. As the functions we are using are probability density functions, Fatou's lemma is handy. It is well known and we omit the proof.

Lemma 6.3 (Fatou's lemma). *Let $f_1, f_2, \ldots$ be a sequence of extended real-valued measurable functions with respect to a probability density function ρ and let $f = \limsup_{n\to\infty} f_n$. If there exists a non-negative integrable function g (i.e. g measurable and $\int_\Omega g(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} < \infty$) such that $f_n \le g$ for all n, then*

$$\limsup_{n\to\infty} \int_\Omega f_n(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} \le \int_\Omega f(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x}.$$

We are now in a position to prove convergence of the Kullback–Leibler divergence, given pointwise convergence of the posterior. As uniform convergence of the posterior has been studied extensively in Section 6.3.3, assuming pointwise convergence is not a

restriction. However, we additionally assume positivity of the posterior, as the Kullback–Leibler divergence becomes undefined otherwise.

Theorem 6.4. *Suppose* $\|q_N(\mathbf{x}\,|\,\mathbf{z}) - q(\mathbf{x}\,|\,\mathbf{z})\|_\infty \to 0$ *for* $N \to \infty$, $q_N(\mathbf{x}\,|\,\mathbf{z}) > \varepsilon\, q(\mathbf{x}) > 0$ *for a* $\varepsilon > 0$, *and* $q(\mathbf{x}\,|\,\mathbf{z}) > 0$ *in* Ω. *Then*

$$D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) \to 0, \text{ for } N \to \infty.$$

Proof. The proof consists of combining Lemma 6.2 and 6.3. The result follows from applying Lemma 6.3 with $f_N = \log \frac{q}{q_N}$ and $f = 0$, in conjunction with $g = \sup_N \log \frac{q}{q_N}$. A direct application of this lemma yields $D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) \to 0$. However, to apply Lemma 6.3, pointwise convergence of $\log \frac{q}{q_N}$ to 0 is necessary. This can easily be seen by applying Lemma 6.2, with $g_N = q_N$ and $g = q$. ■

In a similar way, convergence of $D_{\mathrm{KL}}\big(q_N(\mathbf{x}\,|\,\mathbf{z}) \parallel q(\mathbf{x}\,|\,\mathbf{z})\big) \to 0$ can be proved.

Convergence rate of the Kullback–Leibler divergence

Theorem 6.4 only demonstrates convergence of the Kullback–Leibler divergence. The exact rate of convergence (or the constant involved in it) has not been deduced. In this section we will prove that the convergence rate doubles under mild assumptions, which are:

1. $\|q_N(\mathbf{x}\,|\,\mathbf{z}) - q(\mathbf{x}\,|\,\mathbf{z})\|_\infty \le CQ_N$, with $Q_N \to 0$ for $N \to \infty$,
2. $q_N(\mathbf{z}\,|\,\mathbf{x}) > 0$ and $q(\mathbf{z}\,|\,\mathbf{x}) > 0$,
3. $q(\mathbf{x})$ has compact support.

The first assumption states that the estimated posterior converges, which can be shown using Theorem 6.1. For example, $Q_N = N^{-\alpha}$ if q_N converges algebraically to q with rate α. Many statistical models (such as the model from (6.1) with a uniform prior) fit in these assumptions. Priors on an unbounded domain (e.g. the improper uniform prior or Jackson's prior) cannot be used in the setting of this section due to the last assumption. The convergence result reads as follows.

Theorem 6.5. *Suppose Assumptions 1, 2, and 3 hold.*
Then

$$D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) \le KQ_N^2.$$

Proof. The Kullback–Leibler divergence is always positive, hence

$$\begin{aligned}
D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) &\le D_{\mathrm{KL}}\big(q_N(\mathbf{x}\,|\,\mathbf{z}) \parallel q(\mathbf{x}\,|\,\mathbf{z})\big) + D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) \\
&= \int_\Omega q_N(\mathbf{x}\,|\,\mathbf{z}) \log \frac{q_N(\mathbf{x}\,|\,\mathbf{z})}{q(\mathbf{x}\,|\,\mathbf{z})}\,\mathrm{d}\mathbf{x} + \int_\Omega q(\mathbf{x}\,|\,\mathbf{z}) \log \frac{q(\mathbf{x}\,|\,\mathbf{z})}{q_N(\mathbf{x}\,|\,\mathbf{z})}\,\mathrm{d}\mathbf{x} \\
&= \int_\Omega \big(q_N(\mathbf{x}\,|\,\mathbf{z}) - q(\mathbf{x}\,|\,\mathbf{z})\big) \log\left(\frac{\gamma}{\gamma_N}\,\frac{q_N(\mathbf{z}\,|\,\mathbf{x})}{q(\mathbf{z}\,|\,\mathbf{x})}\right)\mathrm{d}\mathbf{x},
\end{aligned}$$

with γ and γ_N according to (6.18). By taking the absolute value of the integral, we can bound it using the ∞-norm as follows:

$$\begin{aligned} D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) &\leq \int_\Omega \big|q_N(\mathbf{x}\,|\,\mathbf{z}) - q(\mathbf{x}\,|\,\mathbf{z})\big| \left|\log\left(\frac{\gamma}{\gamma_N}\frac{q_N(\mathbf{z}\,|\,\mathbf{x})}{q(\mathbf{z}\,|\,\mathbf{x})}\right)\right| \mathrm{d}\mathbf{x} \\ &\leq \left\|\log\left(\frac{\gamma}{\gamma_N}\frac{q_N(\mathbf{z}\,|\,\mathbf{x})}{q(\mathbf{z}\,|\,\mathbf{x})}\right)\right\|_\infty \int_\Omega \big|q_N(\mathbf{x}\,|\,\mathbf{z}) - q(\mathbf{x}\,|\,\mathbf{z})\big| \,\mathrm{d}\mathbf{x}. \end{aligned}$$

Then, by simplifying the first part of this expression the following is obtained:

$$\begin{aligned} D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) &\leq \|\log(\gamma) - \log(\gamma_N) + \log(q_N(\mathbf{z}\,|\,\mathbf{x})) - \log(q(\mathbf{z}\,|\,\mathbf{x}))\|_\infty \\ &\quad \cdot \int_\Omega \big|q_N(\mathbf{z}\,|\,\mathbf{x}) - q(\mathbf{z}\,|\,\mathbf{x})\big|\, q(\mathbf{x})\,\mathrm{d}\mathbf{x}. \end{aligned}$$

The first term can be bounded using the triangle inequality and the second term can be bounded by using that Ω is bounded, obtaining the following:

$$\begin{aligned} D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) &\leq \big(\|\log(\gamma) - \log(\gamma_N)\|_\infty + \|\log(q_N(\mathbf{z}\,|\,\mathbf{x})) - \log(q(\mathbf{z}\,|\,\mathbf{x}))\|_\infty\big) \\ &\quad \cdot \|q_N(\mathbf{z}\,|\,\mathbf{x}) - q(\mathbf{z}\,|\,\mathbf{x})\|_\infty. \end{aligned}$$

As $\gamma > 0$, the first term can be bounded using the Lipschitz continuity of the logarithm. Moreover, Ω is compact, hence there exists an $A > 0$ such that $q(\mathbf{z}\,|\,\mathbf{x}) > A$ with $\mathbf{x} \in \Omega$. Therefore the second term can also be bounded using the Lipschitz continuity of the logarithm. The last term is already in the right format, and we obtain

$$\begin{aligned} D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) &\leq \frac{1}{|\log A|}\big(\|\gamma - \gamma_N\|_\infty + \|q_N(\mathbf{z}\,|\,\mathbf{x}) - q(\mathbf{z}\,|\,\mathbf{x})\|_\infty\big) \\ &\quad \cdot \|q_N(\mathbf{z}\,|\,\mathbf{x}) - q(\mathbf{z}\,|\,\mathbf{x})\|_\infty. \end{aligned}$$

Finally, the result readily follows:

$$D_{\mathrm{KL}}\big(q(\mathbf{x}\,|\,\mathbf{z}) \parallel q_N(\mathbf{x}\,|\,\mathbf{z})\big) \leq (C_1 Q_N + C_2 Q_N)\, C_2 Q_N = K Q_N^2. \qquad \blacksquare$$

Note that in a similar manner it can be proved that $D_{\mathrm{KL}}\big(q_N(\mathbf{x}\,|\,\mathbf{z}) \parallel q(\mathbf{x}\,|\,\mathbf{z})\big) \leq K Q_N^2$. If $Q_N = N^{-\alpha}$, then $Q_N^2 = N^{-2\alpha}$, demonstrating that the rate of convergence indeed doubles.

6.4. Numerical examples

Three numerical test cases are employed to demonstrate the performance of our methodology. First, in Section 6.4.1 two explicit test cases are studied, which are cases where an expression for u is known that can be evaluated accurately such that the exact posterior can be determined explicitly. We use these cases to verify the theoretical properties that have been derived in Section 6.3. For sake of comparison, these cases are studied using interpolation based on Clenshaw–Curtis nodes, Leja nodes, and the proposed adaptively weighted Leja nodes.

In Section 6.4.2, we study calibration of the one-dimensional Burgers' equation. As an explicit solution is not available here, we can show the practical purposes of the interpolation procedure to problems that are defined implicitly. We determine the interpolant using Clenshaw–Curtis nodes and weighted Leja nodes.

The third test case consists of the calibration of the closure parameters of the Spalart–Allmaras turbulence model. Here, a single evaluation of the likelihood is computationally expensive as it requires the numerical solution of the Navier–Stokes equations. In this case straightforward methods (such as MCMC) become intractable and therefore we will only study weighted Leja nodes. This case is considered in Section 6.4.3.

It is important to emphasize that the interest is in obtaining an accurate estimation of the posterior. Therefore we mainly study the convergence of the posterior and focus to a lesser extent on the convergence of the model.

6.4.1. Explicit test cases

We consider two analytic functions to demonstrate the applicability of the approach. Both functions are analytic in their domain of definition, but one of the two functions cannot be represented globally by means of a power series expansion (which is often challenging in interpolation problems). The first function, a Gaussian function, has a large radius of convergence, such that a single power series expansion can be used to globally approximate the function accurately. The second function, a multivariate extension of Runge's function, yields a power series expansion with a small radius of convergence such that a single power series expansion cannot be used to globally approximate the function. Both functions are defined for any dimension d.

Gaussian function

A well-known class of analytic functions is formed by Gaussian functions. We will use the fourth Genz function (see Table 4.1 on page 89) in the following form to represent the model:

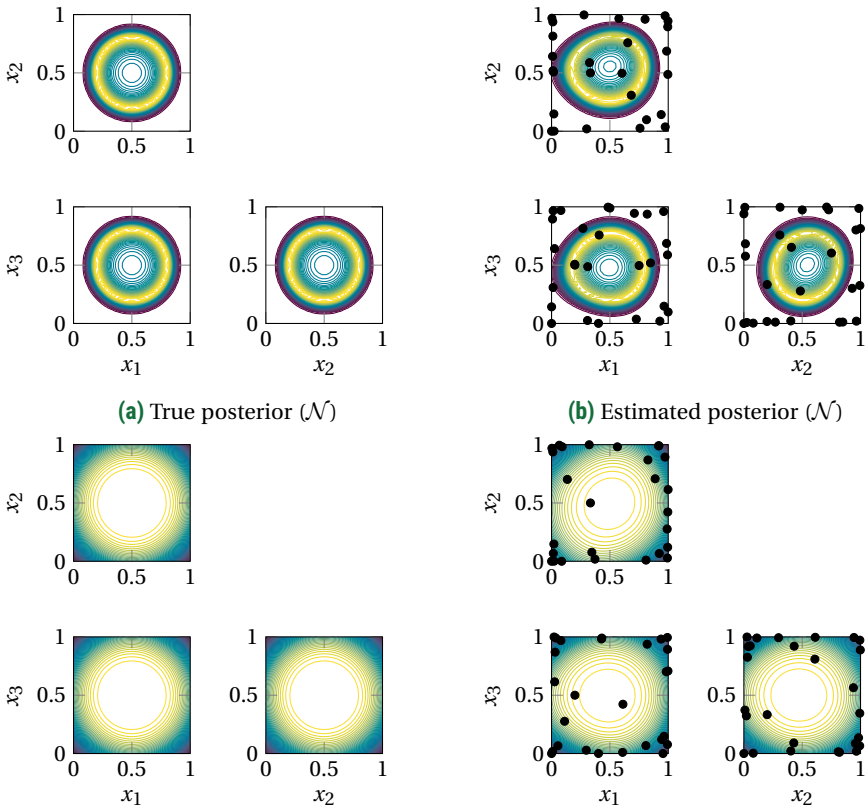
$$u_d\colon [0,1]^d \to \mathbb{R}, \text{ with } u_d(\mathbf{x}) := \exp\left(-\sum_{k=1}^{d}\left(x_k - \frac{1}{2}\right)^2\right).$$

This function is a composition of the exponential function and a polynomial, which are both globally analytic. Hence also this function is globally analytic and can therefore be approximated well using polynomials. Consequently, any nodal set can be used to interpolate this function—even an equidistant set—so we use this test case merely for a sanity check of the procedure and the theory.

Two statistical models are employed to demonstrate the independence of our procedure from the likelihood. The first is the statistical model discussed before, i.e. $z_k = u_d(\mathbf{x}) + \varepsilon_k$, with $\varepsilon_k \sim \mathcal{N}(0, \sigma^2)$ and $\sigma = 1/10$. As discussed before, the likelihood of this statistical model is as follows:

$$q_{\mathcal{N}}(\mathbf{z} \mid \mathbf{x}) \propto \exp\left[-\frac{1}{2}\frac{\|\mathbf{z} - u_d(\mathbf{x})\|_2^2}{\sigma^2}\right],$$

where $\mathbf{z}$ is the vector containing the data. A data vector of $n = 20$ elements is generated by sampling from a Gaussian distribution with mean $u_d(1/4)$ and standard deviation



(a) True posterior ($\mathcal{N}$)

(b) Estimated posterior ($\mathcal{N}$)

(c) True posterior (β)

(d) Estimated posterior (β)

Figure 6.7: True and estimated posteriors with the analytic Gauss function as model using the multivariate normal ($\mathcal{N}$) likelihood and the Beta (β) likelihood with 25 Leja nodes.

σ. The subscript $\mathcal{N}$ refers to multivariate normal. For the second model, we do not write an explicit relation between the data and the model, but simply use the following likelihood:

$$q_\beta(\mathbf{z} \mid \mathbf{x}) \propto \begin{cases} (1-e)^2(1+e)^2 & \text{if } |e| < 1, \\ 0 & \text{otherwise,} \end{cases}$$

where e and $\overline{z}$ are as follows:

$$e = \frac{\overline{z} - u_d(\mathbf{x})}{\overline{z}} \text{ and } \overline{z} = \frac{1}{n}\sum_{k=1}^{n} z_k.$$

We call this likelihood the Beta likelihood (denoted with β), which we use because it has different characteristics than the Gaussian likelihood. As the standard deviation is significantly larger in the second case, the posteriors differ considerably. Both likelihoods are

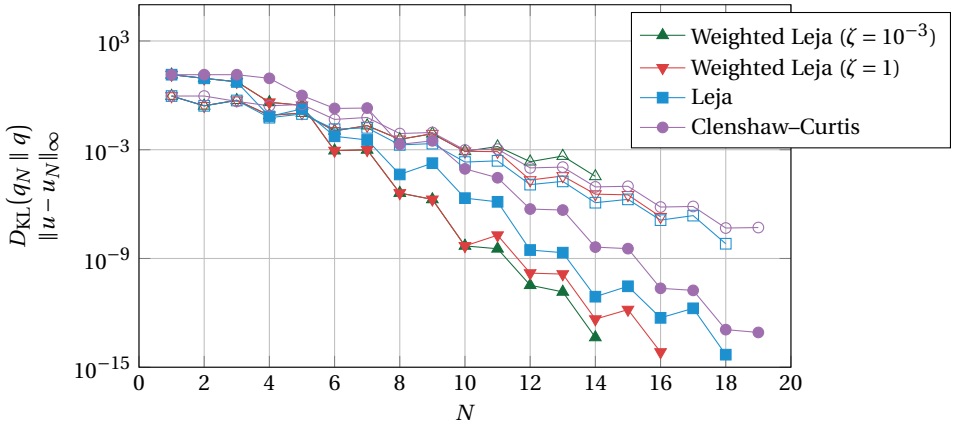


Figure 6.8: Convergence of the interpolant (indicated using open markers) and convergence of the Kullback–Leibler divergence (indicated using filled markers) when using the Gaussian function as model.

continuously differentiable, so we expect similar accuracy when applying the proposed algorithm. In both cases the prior is assumed to be uniform in the domain $[0,1]^d$.

Because the model under consideration is analytic, the value of ζ is not very important for the accuracy of the interpolation procedure (even $\zeta = 0$ works well in this case). We choose $\zeta = 10^{-3}$, because then convergence of the posterior can be observed well, which is shown in Figure 6.7 for $d = 3$. It is clearly visible that for the multivariate normal case, the nodes are placed more in the center of the domain (see for comparison Figure 6.2a). This is also true for the second case, but less apparent due to the less intuitive structure of the posterior. The asymmetry between dimensions occurs due to the interpolation with Leja nodes, which are asymmetric by construction.

If we restrict ourselves to a one-dimensional case, the convergence of our algorithm can be assessed with high accuracy as it is possible to determine the Kullback–Leibler divergence with high accuracy using a quadrature rule. Moreover, a comparison with an interpolant based on Clenshaw–Curtis nodes can be performed. In multivariate cases such a comparison is not feasible, as determining the Kullback–Leibler divergence with high accuracy is intractable either with Monte Carlo (due to the relatively slow convergence) or with a quadrature rule (due to the deterioration of the high accuracy of the univariate quadrature rule). Nonetheless, even in univariate cases the decay of the Kullback–Leibler divergence is relevant for this chapter, as the goal is to construct an accurate *posterior* (and not necessary an accurate *interpolant*, for which various more efficient multivariate interpolation techniques exist).

The Kullback–Leibler divergence cannot be determined for the Beta likelihood. This is because two interpolants u_{N_1} and u_{N_2} in general do not have $|e| < 1$ exactly at the same $\mathbf{x}$. Therefore the set where one model has $e = 0$ and the other $e > 0$ (and vice versa) is measurable, rendering the Kullback–Leibler divergence unbounded.

The Kullback–Leibler divergence is determined using posteriors constructed with

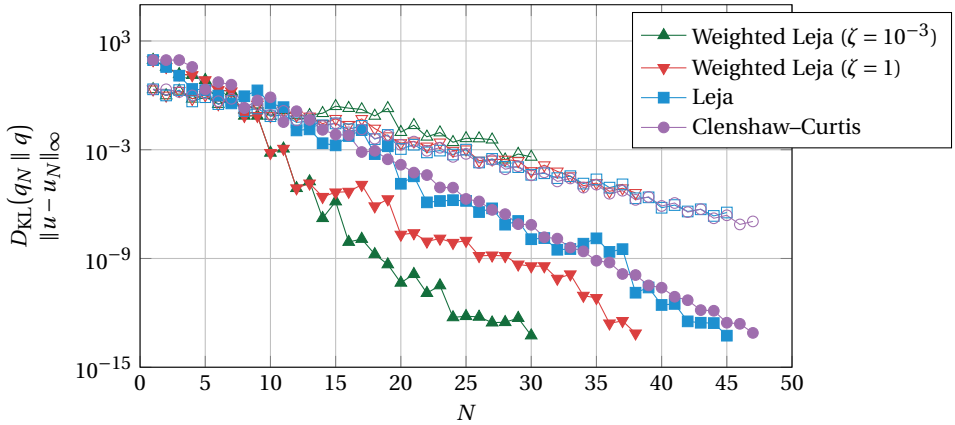


Figure 6.9: Convergence of the interpolant (indicated using open markers) and convergence of the Kullback–Leibler divergence (indicated using filled markers) when using Runge's function as model.

weighted Leja nodes, unweighted Leja nodes, and the Clenshaw–Curtis nodes (see Figure 6.8). All nodal sets under consideration yield a model and a posterior that converge to respectively the true model and true posterior. The doubling of the convergence rate, according to Theorem 6.5, already becomes apparent for the small number of nodes used and it is clearly visible that the weighted Leja nodes are mainly constructed for accuracy of the posterior instead of the model. The weighted Leja nodes with the smallest value of ζ show the fastest convergence, but the difference of model evaluations necessary to reach a certain accuracy level is small as the model under consideration is analytic.

Runge's function

A well-known test case for interpolation methods is the univariate Runge's function. We consider a multivariate extension of it, defined (up to a constant) in the domain $[0,1]^d$ as follows:

$$u_d \colon [0,1]^d \to \mathbb{R}, \text{ with } u_d(\mathbf{x}) = \frac{5}{2 + 50\sum_{k=1}^{d} (x_k - \frac{1}{2})^2}.$$

This function is infinitely smooth, i.e. all derivatives exist and are continuous. However, its Taylor series has small radius of convergence. If a nodal set with exponentially growing Lebesgue constant is used to interpolate this function, the convergence rate is significantly deteriorated (if it converges at all). This is well known and typically called Runge's phenomenon. The Gaussian statistical model, the uniform prior, and the true value $\mathbf{x} \equiv \frac{1}{4}$ are reconsidered for this test case. Results in terms of convergence using the three different nodal sets discussed previously are shown in Figure 6.9 (again $d = 1$).

Differences between the nodal sets are more apparent than in the previous test case. In this case it can be clearly observed that the weighted Leja nodes are mainly constructed to obtain an accurate posterior and that less emphasis is put on the accuracy

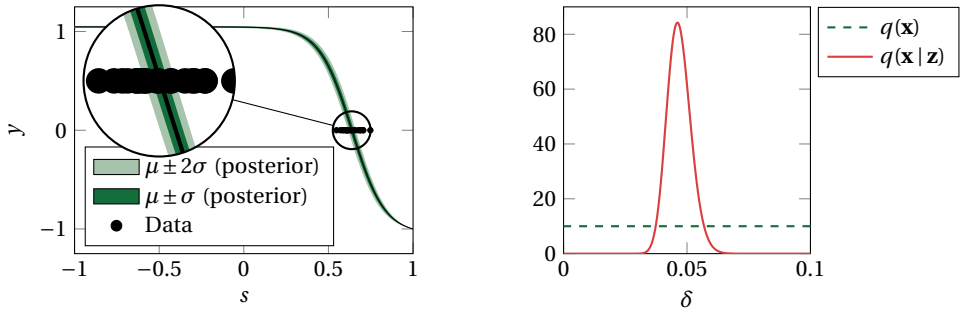


Figure 6.10: *Left:* Propagation of the posterior through the Burgers' equations. *Right:* The prior (dashed) and the posterior of the Burgers' equation calibration test case.

of the interpolant. The weighted Leja nodes outperform the other nodal sets in the Kullback–Leibler divergence. The unweighted Leja nodes and Clenshaw–Curtis nodes perform equally well, which is surprising as the Leja nodes do not have a logarithmically growing Lebesgue constant (contrary to the Clenshaw–Curtis nodes). Figure 6.9 also shows that upon increasing the value of ζ from 10^{-3} to 1, the convergence rate of the weighted Leja sequence slightly decreases. For small N, the posterior dominates the weighting function in Leja nodes, while for large N the effect of ζ becomes important.

6.4.2. Burgers' equation

In this section the Burgers' equation is considered where the boundary condition is unknown, extending the example from Marzouk and Xiu [121]. The one-dimensional steady state Burgers' equation for a solution field $y(s)$ and diffusion coefficient ν is stated as follows:

$$\nu \frac{\partial^2 y}{\partial s^2} - y \frac{\partial y}{\partial s} = 0,$$

where boundary conditions complete the system. In this section, we will consider the equation on an interval $[-1,1]$ with $\nu = 1/10$ and use the boundary conditions $y(-1) = 1+\delta$ and $y(1) = -1$, where δ is unknown.

The inverse problem is as follows. Let s_0 be the zero-crossing of the solution $y(s)$, i.e. $y(s_0) = 0$. Given a vector of noisy observations of s_0, we would like to obtain the value of δ, i.e. reconstruct the boundary conditions. Hence, using the notation of this thesis, we have a function $u\colon \mathbb{R} \to \mathbb{R}$ with $u(\delta) = s_0$. This function maps a boundary condition δ to the zero-crossing of the solution y. In the current setting, this function is only defined implicitly.

Let $\mathbf{z}$ be a vector with $n = 20$ noisy observations of s_0. We generate this vector by sampling δ from a normal distribution with mean $1/20$ and standard deviation $\sigma = 1/20$ and determining the corresponding values of s_0. We use a uniform prior such that $\delta \sim U[0, 1/10]$. The likelihood is Gaussian with zero mean and standard deviation σ, i.e. for each "measurement" z_k (for $k = 1, \ldots, n$) we have

$$z_k - u(\delta) \sim \mathcal{N}(0, \sigma^2).$$

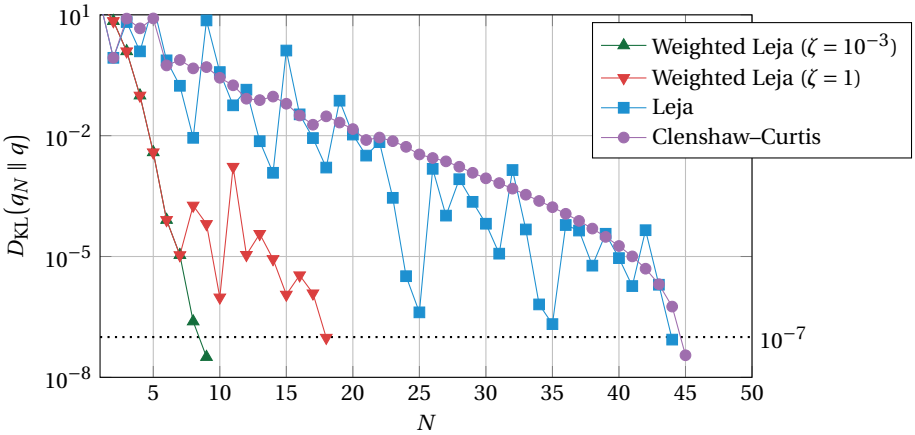


Figure 6.11: Convergence of the posterior of Burgers' equation using three different sampling procedures.

A second-order finite volume scheme is employed to numerically solve the Burgers' equation equipped with a uniform mesh of 10^5 cells. The zero-crossing is determined by piecewise interpolation of the solution.

We use Clenshaw–Curtis (scaled to $[0, 1/10]$), Leja, and weighted Leja nodes to compute the posterior. In all three cases, the interpolant was refined until the Kullback–Leibler divergence of the estimated posterior with respect to the true posterior (computed using a fine quadrature rule) is smaller than 10^{-7}. In Figure 6.10 a sketch of the distribution of u after propagation of this posterior and the exact posterior is depicted. The standard deviation of the output variable u is small because we only take measurement errors into account (hence the standard deviation decreases for larger data sets). The posteriors determined after convergence of the three nodal sequences did not differ visually from the true posterior, so we do not present them separately.

There are large differences in the number of nodes that are necessary to obtain a converged posterior, see Figure 6.11. For the posteriors determined using Clenshaw–Curtis nodes and Leja nodes, approximately 45 nodes are necessary. For the weighted Leja nodes, just 18 nodes are necessary for $\zeta = 1$ and only 9 nodes are necessary for $\zeta = 10^{-3}$. This significant improvement is because the initial nodes already provide a good approximation of the posterior, which is leveraged when ζ is small. Again it is clearly visible that for small N, the posterior dominates the weighting function, while for large N the value of ζ dominates. In both cases the weighted Leja nodes clearly outperform the other nodal sets.

6.4.3. Turbulence model closure parameters

In this section the flow around an airfoil (i.e. the two-dimensional cross section of a wing) is considered. The airfoil under consideration is the RAE2822, because extensive publicly available measurements have been performed for this particular airfoil. We use the wind tunnel measurements of the pressure coefficient from Cook et al. [36]

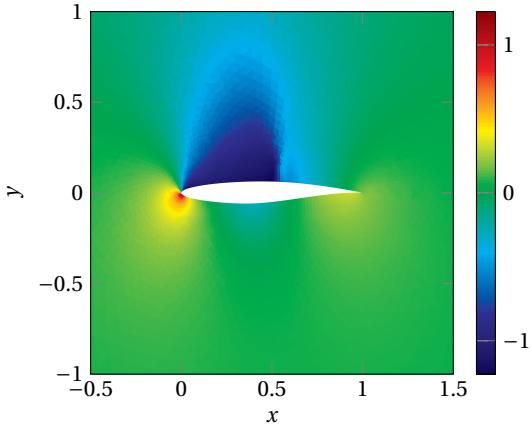


Figure 6.12: The pressure coefficient around the RAE2822 airfoil using the canonical turbulence coefficients.

and study Case 6, i.e. the angle of attack equals 2.92°, the Mach number equals 0.725, and the Reynolds number equals $6.5 \cdot 10^6$. These parameters are not corrected for wind tunnel effects, which is necessary for comparison with numerical simulations. See Slater et al. [168] for more information about such a correction procedure and Cook et al. [36] for more information about the measurement data under consideration, as these details are out of the scope of this chapter.

The key difference between the case studied here and the airfoil flow studied in Chapter 4 is that the flow is transonic, i.e. there is both a subsonic and supersonic region. These type of flows usually have shock formation, which is not the case for the flow studied in Chapter 4. The flow around the airfoil studied in this chapter is depicted in Figure 6.12, determined numerically using the parameters from Case 6 and the canonical turbulence coefficients of the Spalart–Allmaras turbulence model. It is clearly visible that there is shock formation on the upper side of the airfoil.

Problem description

The flow around an airfoil is often modeled using the Navier–Stokes equations. Due to the large range of scales present in the solution of these equations, typically the Reynolds-averaged Navier–Stokes (RANS) equations are employed. The details are out of the scope of this chapter. The interested reader is referred to Wilcox [192]. The RANS equations do not form a closed system of differential equations and require a closure model. Typically, the Boussinesq hypothesis is used, which introduces an eddy viscosity. This viscosity models the effect of turbulence in the flow and requires an additional set of equations called a turbulence model. Throughout the years, many different turbulence models have been developed. These models have fitting parameters, chosen such that the model represents idealized test cases well. Often the coefficients are calibrated in a non-systematic way (e.g. by hand). A turbulence model that is commonly used for

flow over airfoils is the Spalart–Allmaras turbulence model, which consists of a single equation modeling the transport, production, and dissipation of the eddy viscosity [170].

Several forms of this model exist. The original Spalart–Allmaras turbulence model has 9 constants, defined as follows:

σ	2/3	C_{b1}	0.1355	C_{b2}	0.622
κ	0.41	C_{w2}	0.3	C_{w3}	2.0
C_{t3}	1.2	C_{t4}	0.5	C_{v1}	7.1

Here we omit C_{w1}, which is commonly defined as $C_{w1} = C_{b1}/\kappa^2 + (1 + C_{b2})/\sigma$.

Although tested extensively, straightforward physical interpretations do not exist for many of the parameters above. In this section, we calibrate these parameters systematically using Bayesian model calibration and apply the weighted Leja nodes proposed in this chapter. We use $\zeta = 1$, to make sure that we have convergence without spurious oscillations. This value is probably slightly too large than necessary to obtain a converging approximate posterior, but due to the large computational expense that is necessary for these simulations and the inability to run these simulations a large number of times we choose ζ rather too large than too small. Results on the calibration of turbulence parameters exist in literature [34, 51], where it is customary to calibrate the parameters using MCMC methods.

The computer code `SU2` is employed to numerically solve the RANS equations, which is a second-order finite-volume computational fluid dynamics solver with support for the Spalart-Allmaras turbulence model. We have made a few minor modifications to allow the turbulence parameters to be configurable. `SU2` has been tested extensively to the airfoil test case (for the canonical turbulence parameters) with these options, see Palacios et al. [139].

6

Statistical model

We consider the same parameters for calibration as Edeling et al. [51] and Cheung et al. [34], namely $\boldsymbol{\vartheta} = (\kappa, \sigma, C_{b1}, C_{b2}, C_{v1}, C_{w2}, C_{w3})^{\mathrm{T}}$. The remaining parameters are fixed at their canonical values. The parameters are denoted using $\boldsymbol{\vartheta}$ (instead of $\mathbf{x}$) to avoid confusion with the commonly used nomenclature for the spatial parameters.

We are now in the position to state the statistical model. Let s be the spatial parameter along the contour of the airfoil that runs from the tip of the airfoil in anticlockwise direction over the airfoil, i.e. s parametrizes the airfoil such that each s has a unique value for the pressure coefficient. Suppose the data (i.e. a vector of pressure coefficients) $\mathbf{z} = (z_1, \ldots, z_n)^{\mathrm{T}}$ is provided at locations $\mathbf{s} = (s_1, \ldots, s_n)^{\mathrm{T}}$ on the airfoil. Then the statistical model under consideration is as follows:

$$\begin{aligned} v(\boldsymbol{\vartheta}; s_k) &= u(\boldsymbol{\vartheta}; s_k) + \delta(s_k), \\ z_k &= v(\boldsymbol{\vartheta}; s_k) + \varepsilon_k, \end{aligned}$$

where $\delta \sim \mathcal{N}(0, \mathrm{Cov}(s, s'))$ is a Gaussian process with mean 0 and squared exponential covariance function

$$\mathrm{Cov}(s, s') = A \exp\left[-\frac{(s - s')^2}{2l^2}\right].$$

Moreover, we choose all ε_k to be independently and identically Gaussian distributed with mean zero and standard deviation $\tilde{\sigma}$, which is defined explicitly later (we use $\tilde{\sigma}$ to avoid confusion with one of the turbulence coefficients). The measurement data from Cook et al. [36] consists of $n = 103$ measurements obtained on the surface of the airfoil.

We choose the model such that v is the "true" process that is modeled by u. The random process δ and random variables ε_k represent the model and measurement error respectively. Both require an estimation based on knowledge of the model and the data, as our calibration framework currently does not include hyperparameters. Bounds for the measurement error are provided in Cook et al. [36], resulting in $\tilde{\sigma} = 0.01$. This value is larger than the error of the measurement data, thus incorporating potential additional errors. The parameters of the Gaussian process are chosen to be $A = 0.6$ and $l = 0.2$, based on the largest error that we expect (which is approximately 0.5, based on the strength of the shock on top of the airfoil) and the distance between two measurement locations.

Prior and likelihood

The prior is assumed to be multivariate uniformly distributed with mean equal to the canonical values of the parameters and standard deviation that encapsulates a large number of variants of the Spalart–Allmaras turbulence models. More specifically, we choose the support of the prior as follows:

κ	$[0.205, 0.615]$
σ	$[1/3, 1]$
C_{b1}	$[0.0678, 0.2033]$
C_{b2}	$[0.311, 0.933]$
C_{v1}	$[3.55, 10.65]$
C_{w2}	$[0.2, 0.4]$
C_{w3}	$[1, 3]$

The likelihood readily follows from the statistical model:

$$q(\mathbf{z} \mid \boldsymbol{\vartheta}) \propto \exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}} C^{-1} \mathbf{d}\right],$$

with $\mathbf{d}$ the misfit and C the covariance, i.e.

$$\begin{aligned} d_k &= z_k - u(\boldsymbol{\vartheta}; s_k), \\ C &= \tilde{\sigma} I + \Sigma, \text{ with } \Sigma_{i,j} = \mathrm{Cov}(s_i, s_j). \end{aligned}$$

Finally, the posterior is formed by a multiplication of the likelihood with the prior in the usual way.

Results

The closure coefficients are calibrated using 100 adaptively weighted Leja nodes. The convergence is assessed using cross-validation between two consecutive interpolants,

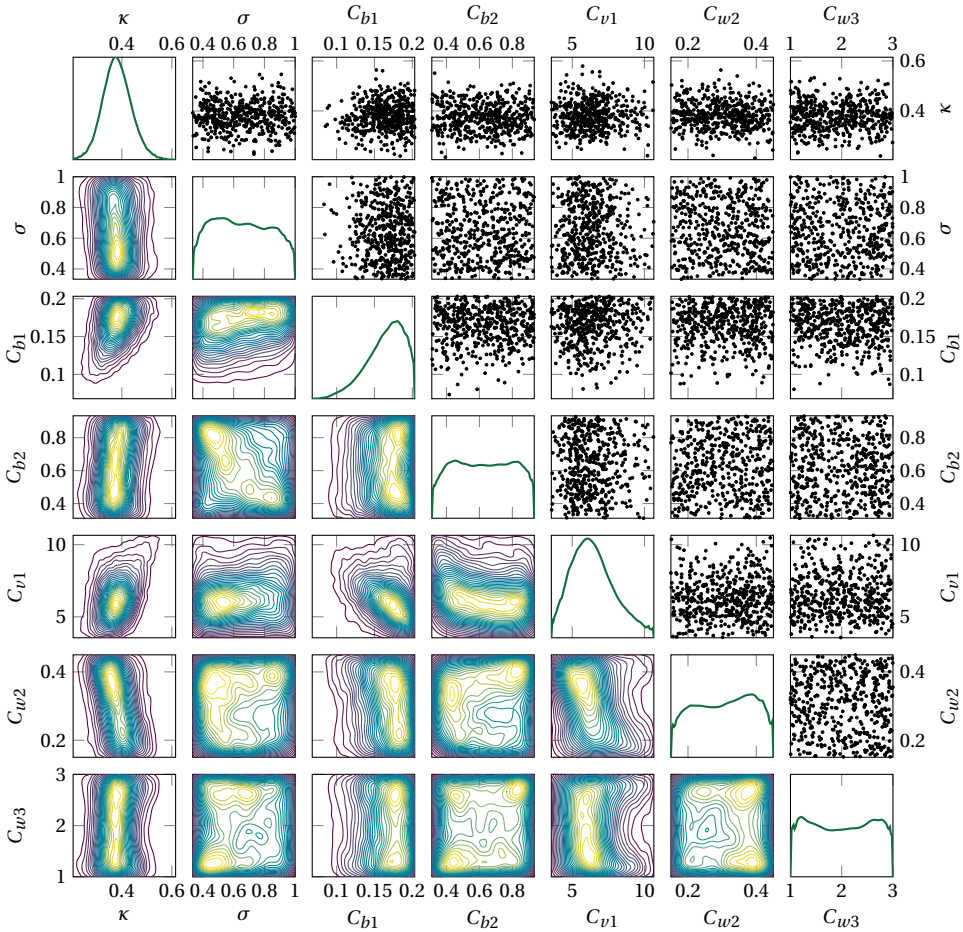


Figure 6.13: The marginals of the posterior under consideration of the RAE2822 airfoil case.

6

up to the point that visually no difference can be observed in the posterior. The one- and two-dimensional marginals from the obtained posterior are depicted in Figure 6.13. We cannot compare the result with the true posterior, as done in the previous two test cases, so instead a qualitative comparison is made with results obtained in literature.

The results are similar to Cheung et al. [34], who calibrated the same set of parameters using MCMC using more than 30 000 samples. In their study, the parameters κ and C_{v1} are informed best, which is also clearly visible in Figure 6.13. In our results, the parameter C_{b1} is also informed in comparison with the other parameters, but this is not the case in the study from Cheung et al. There can be various reasons for this, among others the fact that Cheung et al. used a flat plate test case for the calibration and used the skin friction coefficient as data.

In the study from Edeling et al. [51] a similar test case is performed. Based on a sensitivity analysis, it follows that the parameters κ and C_{v1} have the highest Sobol' indices and are therefore most influential on the model output. This is consistent with our study, since these parameters are informed best, and with the study from Cheung et al. The results for C_{b1} again differ, which we attribute to the fact that Edeling et al. also used the skin friction coefficient and the flat plate test case.

To conclude, the proposed method is capable of constructing a good approximate posterior using a fraction of the 30 000 model evaluations that were necessary in the MCMC case. The posterior compares qualitatively very well with both studies. By applying MCMC to the constructed surrogate, we can propagate the posterior and make a prediction under uncertainty. To illustrate this, the posterior is propagated through the surrogate to obtain uncertainty bounds on the pressure coefficient. The results are depicted in Figure 6.14. It is clearly visible that the largest uncertainty originating from the posterior is near the shock on top of the airfoil. This is studied more extensively in Chapter 7.

6.5. Conclusion

Bayesian model calibration is an attractive approach for calibrating model parameters of complex physical models. It is customary to sample the posterior using MCMC methods, but these are only tractable if the model under consideration can be evaluated rapidly. We consider problems where this is not the case, such as the calibration of turbulence closure coefficients.

Our proposed method consists of replacing the computationally expensive model with an interpolating surrogate model. The interpolant is determined adaptively using weighted Leja nodes, where the weighting function directly uses the posterior and is changed with each iteration. We have proposed a formulation in which the weighting function consists of two parts: a part incorporating the currently available posterior and a part incorporating the accuracy of the interpolant. The balance between these two can be changed adaptively by the parameter ζ and convergence is guaranteed for any $\zeta > 0$. The "best" value of ζ depends on the specifics of the model, the likelihood, and the posterior, and can be changed adaptively during the calibration procedure. Compared to conventional nodal sets (such as Clenshaw–Curtis nodes), we obtain more accurate results with less nodes.

Theoretically we have proved that if the interpolant converges to the true model

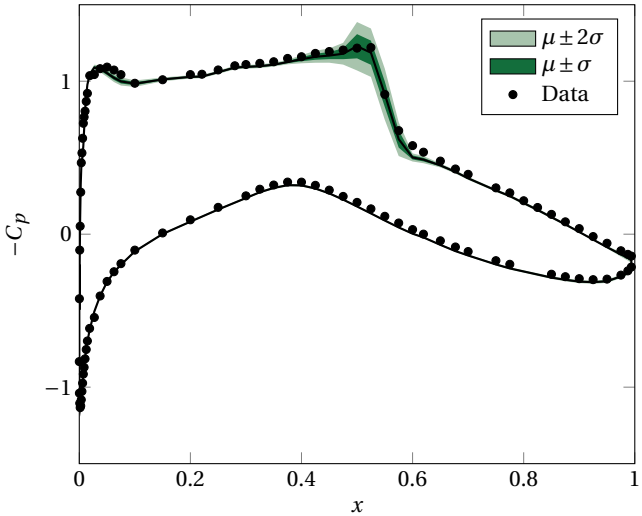


Figure 6.14: Pressure coefficient along the airfoil determined by propagation of the posterior through the interpolating surrogate. The mean μ is depicted as a solid line and the standard deviation σ is shaded in green.

in the ∞-norm, then the estimated posterior converges to the true posterior in the ∞-norm with a similar rate. Under mild conditions, the Kullback–Leibler divergence between the estimated and exact posterior converges with a doubled rate.

The three conducted numerical experiments confirm these theoretical findings: if the interpolant converges to the model, the Kullback–Leibler divergence converges to zero with doubled rate. For the explicit numerical test cases, this doubled convergence in the Kullback–Leibler divergence was clearly visible. The calibration of the Burgers' equation shows that the approach also works well for models that are defined implicitly.

Finally, calibration of turbulence closure parameters has been discussed, showing that the approach is truly applicable to computationally expensive models. We have compared the results from our calibration procedure with results conducted using Monte Carlo methods and the posteriors show good resemblance, at a highly reduced computational cost.

There are various approaches to further extend or improve the proposed framework. There are some limitations about the statistical model that can be used in our approach, e.g. we require that it can be written as a function of the model and no hyperparameters are used in the statistical model. Extending the current approach to such a setting is an important step, as hyperparameters are often employed in Bayesian model calibration. To this end, an alternative approach that incorporates hyperparameters and is based on quadrature rules is discussed in Chapter 7.

Secondly, we believe the value of ζ should depend on the specifics of the model and further research is required to find the optimal value either a priori or dynamically during the procedure.

7

Bayesian prediction with the implicit quadrature rule

Bayesian model calibration is a flexible approach to calibrate computational models using measurement data of an output of the model. As discussed extensively in Chapter 6, the result of the calibration is a posterior distribution over the parameters of a computational model. If the posterior is known, it can be used in conjunction with the statistical model to predict new observations. These predictions incorporate all uncertainties, biases, and errors that are present in the statistical model. Inferring predictions in this way is often called Bayesian prediction. Similarly to Bayesian model calibration this is a computationally costly procedure, which is the key focus of this chapter.

As demonstrated in the previous chapter, a viable approach to alleviate the high computational cost associated with sampling from the posterior is to replace the computationally complex model with an interpolating polynomial and use the polynomial as replacement of the model. However, for the purpose of Bayesian prediction the interest is neither in approximating the posterior nor in approximating the model, but solely in computing integral quantities weighted with the posterior. For this purpose, it is a natural idea to employ the quadrature rules as introduced in Chapter 3 and Chapter 4. All numerical estimations computed using these rules are numerically stable, provided that the rules accurately integrate polynomials and have positive weights. This is in contrast to interpolation, where the stability of the interpolant is highly sensitive to the nodal set.

In this chapter, the implicit quadrature rule as introduced in Chapter 4 is employed to construct quadrature rules weighted with arbitrary distributions. The posterior is incorporated by interpolating all available point evaluations using nearest neighbor interpolation. The obtained iterative approach yields a sequence of nested quadrature rules with positive weights that converge to a quadrature rule that computes integrals

---

This chapter is based on the following article: L. M. M. van den Bos, B. Sanderse, and W. A. A. M. Bierbooms. Adaptive sampling-based quadrature rules for efficient Bayesian prediction. *Under review*, 2019. arXiv: 1907.08418 [math.NA].

weighted with the posterior. Again the Spalart–Allmaras turbulence model is considered to demonstrate the effectiveness of the approach.

7.1. Introduction

Bayesian prediction is essentially equivalent to uncertainty propagation and, as discussed before, often involves computing quantities in the form of integrals with a computationally expensive integrand. Various approaches exist to numerically estimate such integrals, for example Monte Carlo approaches [31] or deterministic quadrature rule approaches [24] as discussed in Chapter 3 and 4. In these approaches it is often assumed that samples from the probability distribution can be constructed straightforwardly or are readily available, which is not the case in Bayesian problems [65, 95]. In this case, computing weighted integrals is commonly known as Bayesian prediction and this is the main topic of this chapter.

A commonly used approach to calculate weighted integrals with respect to non-trivial distributions (such as posteriors) is importance sampling [21, 44, 179]. In this approach samples are drawn from a *proposal distribution* and by means of weighting these samples are used to determine integrals with respect to a *target distribution*, i.e. the distribution for which it is difficult to construct samples. If the proposal distribution is chosen well, the constructed samples yield similar convergence rates as samples constructed directly with the target distribution. A major disadvantage is that constructing the proposal distribution is not trivial and often requires a priori knowledge about the target distribution. A well-known extension of importance sampling to a Bayesian framework is formed by Markov chain Monte Carlo methods [80, 122], where the proposal distribution is chosen such that the samples form a Markov chain (as outlined in Section 2.2.2). In this case the proposal distribution is not constructed beforehand, though a large number of costly evaluations of the posterior is typically necessary to construct the sequence of samples. There exist various extensions of Markov chain Monte Carlo methods to reduce the computational effort that is necessary to obtain a burned-in sequence [3, 143, 186, 187].

Even if samples can be drawn from the distribution, the error of estimating an integral by means of random sampling decays as $\mathcal{O}(1/\sqrt{K})$, with K the number of samples (as discussed in Section 2.1.1). Therefore using importance sampling techniques, or Monte Carlo techniques in general, in conjunction with a computationally costly model is often infeasible. In summary, estimating integrals by directly sampling from the posterior is often difficult or very expensive with existing methods.

The slow convergence of random sampling can be alleviated by means of a surrogate, as discussed in Chapter 6. Other well-known approaches include Gaussian process regression [10, 173] and polynomial approximation [11, 121, 159, 161]. If the weighting distribution is not known explicitly, it is difficult to construct an optimal surrogate (and moreover it often requires analyticity or smoothness), so often the surrogate is constructed such that it is globally accurate or the distribution is tempered [107], which yields a similarly accurate surrogate and which has been used in Chapter 6.

At the same time, in the context of Bayesian prediction the interest is not directly in accurately approximating the *model*, but in calculating *moments* of the model with respect to the posterior. These moments are integral quantities, which can be com-

puted more efficiently using a quadrature rule (compared to polynomial interpolation approaches, see Section 2.1.3). Existing quadrature rule approaches [24] and those discussed in this thesis provide rapidly converging estimates of the integral for smooth functions and stable albeit more slowly converging estimates if the integrand is not smooth, see e.g. Figure 3.9 on page 63, Figure 4.8 on page 91, or Figure 5.5 on page 115. However, the rules derived so far in this thesis and existing state-of-the-art quadrature rule approaches such as Gaussian quadrature rules [74], multivariate sparse grids [81, 135, 169], or heuristic optimization approaches [89, 130] require broad knowledge about the distribution under consideration, so these are not directly applicable to the Bayesian setting considered in this chapter.

Therefore, the main goal of this chapter is to propose a novel flexible method for Bayesian prediction that combines the high convergence rates of quadrature rules with the flexibility of adaptive importance sampling. The idea is to construct a sequence of quadrature rules and a sequence of proposal distributions, where the proposal distributions are constructed using all available model evaluations. The sequence of nested quadrature rules converges to a quadrature rule that incorporates the (expensive) probability distribution (i.e. the posterior). By ensuring that all constructed quadrature rules have high degree and have positive weights, high convergence rates are obtained (depending on the smoothness of the integrand). The rules are specifically tailored to determine moments weighted with a posterior and consequently require a small number of nodes to obtain an accurate estimation. Moreover it can be demonstrated rigorously that estimations made with the sequence of quadrature rules converge to the exact integral for a large class of integrands and distributions.

This chapter is structured as follows. In Section 7.2 the problem of Bayesian prediction with a quadrature rule is briefly introduced, with a main focus on quadrature rule methods and their relevant properties. In Section 7.3 the main algorithm of this chapter is discussed: a method to construct a quadrature rule that converges to a quadrature rule of the posterior. The method is analyzed theoretically in Section 7.4 and to demonstrate the applicability of the quadrature rule to computational problems, the technique is applied to some test functions in Section 7.5. Moreover the calibration of the turbulence closure coefficients of the RAE2822 airfoil is discussed. The chapter is summarized and concluded in Section 7.6.

7.2. Preliminaries

Bayesian prediction can be embedded straightforwardly in the Bayesian setting using in this thesis so far. It is introduced in Section 7.2.1, where also the main quantities that are computed in this chapter are introduced. These quantities are computed using a quadrature rule, a concept that has been used frequently in this thesis (e.g. see Section 2.1.3). Since the distribution under consideration is adaptively changed in this chapter, some additional nomenclature is necessary to extend quadrature rules to this setting. This is discussed in Section 7.2.2.

7.2.1. Bayesian prediction

The key goal of this chapter is to propose an efficient method to infer Bayesian predictions involving a computationally expensive model. However, the setting where

the proposed methods can be applied is slightly more general than that. As usual, let $u\colon \Omega \to \mathbb{R}^n$ be a continuous function with $\Omega \subset \mathbb{R}^d$ $(d = 1,2,3,\dots)$ that describes the quantity of interest and let $\rho\colon \Omega \to \mathbb{R}$ be a continuous and bounded probability density function (PDF). Then determining Bayesian predictions can be described as determining the expectation of a random variable $f(\mathbf{X})$, with $f\colon \Omega \to \mathbb{R}$ a function such that $f = F \circ u$ for an explicitly known map F (so F is defined on the *range* of u). Here, $\mathbf{X}$ is a d-variate random variable with PDF ρ. The mean is an integral quantity and will be denoted by the operator $\mathcal{I}^\rho$ throughout this chapter, i.e.

$$\mathcal{I}^\rho f := \mathbb{E}[f(\mathbf{X})] = \int_\Omega f(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_\Omega F(u(\mathbf{x}))\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{7.1}$$

The main difference in nomenclature with previous chapters is that the distribution of the integral is explicitly mentioned in $\mathcal{I}$.

Integrals of this type have been studied many times in the framework of uncertainty propagation and many efficient methods exist to approximate them [105, 174, 201]. The approach taken in this chapter is based on combining the implicit quadrature rule as introduced in Chapter 4 with a nearest neighbor interpolant of $\rho(\mathbf{x})$. Geometric approaches based on nearest interpolation are closely related to the Voronoi tessellation, which have been used successfully to construct estimates of integrals as considered in this thesis [29, 48, 77, 114]. However, in these cases the distribution ρ is known explicitly beforehand (possibly up to a constant). In this chapter, the computationally challenging part is that $\rho(\mathbf{x})$ is obtained by Bayesian analysis, i.e. it is, up to a constant, a *posterior* obtained using Bayes' law:

$$q(\mathbf{x} \mid \mathbf{z}) \propto q(\mathbf{z} \mid \mathbf{x})\, q(\mathbf{x}) =: \rho(\mathbf{x}),$$

where $\mathbf{z}$, $q(\mathbf{x} \mid \mathbf{z})$, $q(\mathbf{z} \mid \mathbf{x})$, and $q(\mathbf{x})$ are the measurement data, the posterior, the likelihood, and the prior respectively. See Section 2.2.1 for a more elaborate introduction. Throughout this chapter we neglect the scaling factor (called the *evidence*) in this definition of ρ, as it is not of importance for the proposed method.

Bayesian prediction consists of assessing the *posterior predictive distribution*, which describes the probability of observing new data $\widehat{\mathbf{z}}$ given the existing data $\mathbf{z}$. It is obtained by marginalizing over the posterior as follows [52]:

$$q(\widehat{\mathbf{z}} \mid \mathbf{z}) = \int_\Omega q(\widehat{\mathbf{z}}, \mathbf{x} \mid \mathbf{z})\,\mathrm{d}\mathbf{x} = \int_\Omega q(\widehat{\mathbf{z}} \mid \mathbf{x}, \mathbf{z})\, q(\mathbf{x} \mid \mathbf{z})\,\mathrm{d}\mathbf{x} = \int_\Omega q(\widehat{\mathbf{z}} \mid \mathbf{x})\, q(\mathbf{x} \mid \mathbf{z})\,\mathrm{d}\mathbf{x}.$$

Here, we assume that $\widehat{\mathbf{z}}$ and $\mathbf{z}$ are conditionally independent given $\mathbf{x}$. Moments of the posterior predictive distribution can be used among others to obtain predictions of unmeasured quantities. These moments can often be expressed explicitly as integrals over the computational model u.

In general we require throughout this chapter that the distribution ρ is such that an evaluation of $\rho(\mathbf{x})$ requires an evaluation of $u(\mathbf{x})$. Moreover if u has been evaluated for a specific value of $\mathbf{x}$, the value of $\rho(\mathbf{x})$ can be determined without any significant computational cost. In other words, there is a function $G\colon \mathbb{R} \to \mathbb{R}$ such that $\rho = G \circ u$. This is often the case if $\rho(\mathbf{x})$ forms a posterior in a Bayesian framework and this property is the key observation that will be leveraged to approximate (7.1).

A well-known example of a prior and a likelihood for a scalar model u are a uniform prior, defined as $q(\mathbf{x}) \propto 1$ for $\mathbf{x} \in \Omega$, and a Gaussian likelihood, defined as follows for given standard deviation σ:

$$q(\mathbf{z} \,|\, \mathbf{x}) \propto \exp\left[-\frac{1}{2}\frac{\sum_{k=1}^{n}(u(\mathbf{x}) - z_k)^2}{\sigma^2}\right], \text{ with measurement data } \mathbf{z} = (z_1, \dots, z_n)^{\mathrm{T}} \text{ known.}$$

In this particular case, the expectation of the posterior predictive distribution can be explicitly expressed as follows:

$$\begin{aligned}\mathbb{E}[\widehat{\mathbf{z}} \,|\, \mathbf{z}] &= \int_Z \widehat{\mathbf{z}}\, q(\widehat{\mathbf{z}} \,|\, \mathbf{z})\, \mathrm{d}\widehat{\mathbf{z}} = \int_\Omega \int_Z \widehat{\mathbf{z}}\, q(\widehat{\mathbf{z}} \,|\, \mathbf{x})\, q(\mathbf{x} \,|\, \mathbf{z})\, \mathrm{d}\widehat{\mathbf{z}}\, \mathrm{d}\mathbf{x} \\ &= \int_\Omega q(\mathbf{x} \,|\, \mathbf{z}) \int_Z \widehat{\mathbf{z}}\, q(\widehat{\mathbf{z}} \,|\, \mathbf{x})\, \mathrm{d}\widehat{\mathbf{z}}\, \mathrm{d}\mathbf{x} = \int_\Omega q(\mathbf{x} \,|\, \mathbf{z})\, \mathbb{E}[\widehat{\mathbf{z}} \,|\, \mathbf{x}]\, \mathrm{d}\mathbf{x} = \int_\Omega q(\mathbf{x} \,|\, \mathbf{z})\, u(\mathbf{x})\, \mathrm{d}\mathbf{x}. \qquad (7.2)\end{aligned}$$

Here, Z denotes the space that contains all possible values of the data. It is not necessary to derive this space formally, as it is only used in an intermediate step of the derivation. Notice that the obtained integral is the expectation of the computational model with the parameters distributed according to the posterior. Uncertainty propagation with the posterior as distribution, as presented for example in Figure 6.14, is therefore a basic example of Bayesian prediction.

Various statistical models that combine measurement error and model error exist for the purpose of Bayesian model calibration. We refer the interested reader to Kennedy and O'Hagan [95] and will consider such an extensive model in a numerical example discussed in Section 7.5. Vector-valued u can be incorporated straightforwardly in the framework of this chapter, since only the posterior distribution $q(\mathbf{x} \,|\, \mathbf{z})$ is used to construct numerical integration techniques.

Throughout this chapter, two assumptions are imposed on the statistical setting. Firstly, we assume that the statistical moments of the prior are finite, i.e.

$$\int_\Omega |\varphi(\mathbf{x})|\, q(\mathbf{x})\, \mathrm{d}\mathbf{x} < \infty, \text{ for all polynomials } \varphi.$$

This implies among others that the prior is not improper. Secondly, it is assumed that the posterior is continuous and bounded. These two assumptions are not restrictive, e.g. statistical models with Gaussian random variables fit in this setting.

These assumptions are only necessary for the theoretical analysis of the approach discussed in this chapter, since the construction of quadrature rules is based on the approximation by means of polynomials. The approach does not require any major modifications in order to apply it to cases where an improper prior or a discontinuous posterior is considered. An example of the latter is discussed in more detail in Section 7.5.1.

7.2.2. Quadrature rules

The integral from (7.1) is approximated by means of a quadrature rule, consisting of nodes $\{\mathbf{x}_k\}_{k=0}^N \subset \Omega$ and weights $\{w_k\}_{k=0}^N \subset \mathbb{R}$:

$$\int_\Omega f(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_{k=0}^{N} f(\mathbf{x}_k)\, w_k. \qquad (7.3)$$

For the purposes of this chapter, it is important to emphasize (again) that the distribution $\rho(\mathbf{x})$ does not appear explicitly on the right-hand side of this equation. We will denote the quadrature rule approximation by means of the linear operator $\mathcal{A}_N^\rho$, i.e.

$$\mathcal{A}_N^\rho f := \sum_{k=0}^{N} f(\mathbf{x}_k)\, w_k.$$

Similarly to the notation of the integral operator, the distribution ρ is explicitly mentioned. If $\rho(\mathbf{x}) = q(\mathbf{z}\,|\,\mathbf{x})\, q(\mathbf{x})$, a quadrature rule with respect to the posterior $q(\mathbf{x}\,|\,\mathbf{z})$ can be obtained by rescaling the weights of a quadrature rule of ρ such that $\sum_{k=0}^{N} w_k = 1$, provided that $\mathcal{A}_N^\rho 1 = \mathcal{I}^\rho 1$.

As discussed extensively before, three properties are relevant for quadrature rules that are applied to computationally expensive models: the exactness of the quadrature rule, positivity of the weights, and nesting. The first two properties ensure numerical stability of the quadrature rule and make that the quadrature rule converges. A nested quadrature rule allows for straightforward refinement of quadrature rule estimations, as computationally costly model evaluations can be reused.

The notation describing these properties is exactly the same as used before in this thesis. To this end, let $\varphi_0, \dots, \varphi_D$ be $D+1$ d-variate polynomials defined on Ω and consider the space $\Phi_D = \operatorname{span}\{\varphi_0, \dots, \varphi_D\}$. Then the goal is to construct a quadrature rule such that it integrates all functions $\varphi \in \Phi_D$ exactly. This is, due to linearity, equivalent to

$$\mathcal{A}_N^\rho \varphi_j = \mathcal{I}^\rho \varphi_j, \text{ for } j = 0, \dots, D. \tag{7.4}$$

Throughout this chapter the polynomials are sorted graded lexicographically. This ensures that referring to φ_j for any j is well defined and that $\Phi_D \subset \Phi_{D+1}$ for any D, but other monomial orders that are a well-order can be used straightforwardly. Moreover we assume that φ_0 is the constant polynomial, as that firstly allows to straightforwardly scale the quadrature rule weights to incorporate the evidence and secondly allows to apply the implicit quadrature rule from Chapter 4.

The close relation between the nodes and the weights of a quadrature rule is described by the $(D+1) \times (N+1)$ *Vandermonde matrix* $V_D(X_N)$, as introduced by (4.4) on page 71. This matrix is obtained by explicitly writing (7.4) as a system of linear equations.

If the weights are non-negative (i.e. $w_k \geq 0$ for all k), the dimension of Φ_D is a measure for the accuracy of the quadrature rule and increasing D yields a converging estimate for sufficiently smooth functions. This can be seen by using the Lebesgue inequality. However, in this chapter it is not obvious that $\|\mathcal{I}^\rho\|_\infty = 1$, so the inequality is used in the following form:

$$|\mathcal{I}^\rho u - \mathcal{A}_N^\rho u| \leq 2\|\mathcal{I}^\rho\|_\infty \inf_{\varphi \in \Phi_D} \|u - \varphi\|_\infty.$$

As explained before, the weights of the quadrature rule can be scaled such that ρ forms a probability density function. In that case, it holds that $\|\mathcal{I}^\rho\|_\infty = \|\mathcal{A}_N^\rho\|_\infty = 1$ and the Lebesgue inequality as introduced in (2.12) (see page 16) can be used:

$$|\mathcal{I}^\rho u - \mathcal{A}_N^\rho u| \leq 2 \inf_{\varphi \in \Phi_D} \|u - \varphi\|_\infty. \tag{7.5}$$

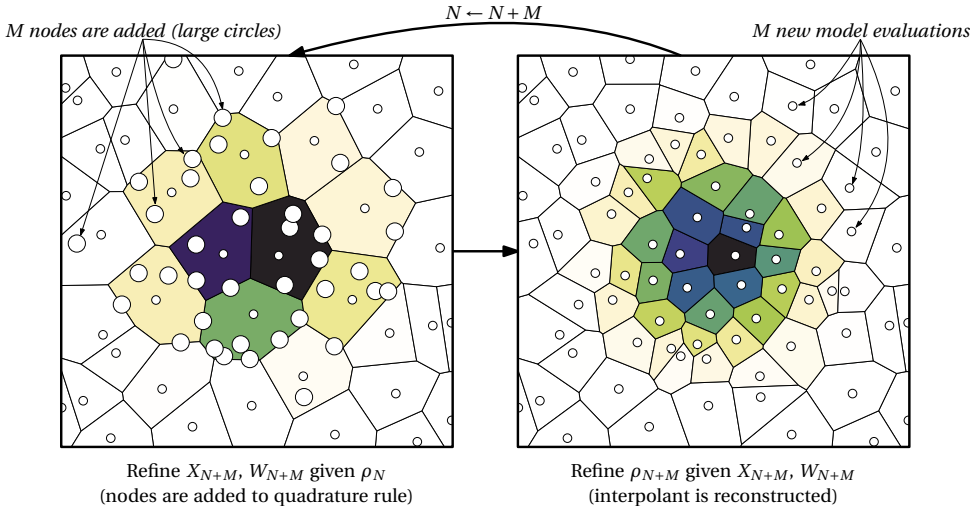


Figure 7.1: Schematic overview of the two main steps of the method to construct the quadrature rules proposed in this chapter. The procedure can be initialized using a single randomly drawn node from the prior, constituting a quadrature rule that consists of a single node with weight equal to unity.

7.3. Bayesian prediction with an adaptive quadrature rule

The main problem addressed in this chapter is accurate numerical estimation of integrals obtained in Bayesian prediction. More specifically, the main interest is the construction of quadrature rules to approximate integrals weighted with the posterior, a problem that was summarized in (7.3) as follows:

$$\mathcal{I}^{\rho} f = \int_{\Omega} f(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_{k=0}^{N} f(\mathbf{x}_k)\, w_k = \mathcal{A}_N^{\rho} f.$$

Here, both f and ρ are functions of u, denoted by $f(\mathbf{x}) = F(u(\mathbf{x}))$ and $\rho(\mathbf{x}) = G(u(\mathbf{x}))$ for suitable F and G. The functions F and G are such that they can be evaluated efficiently and quickly. For example, in (7.2) we have $F(u(\mathbf{x})) = u(\mathbf{x})$ and $G(u(\mathbf{x})) = q(\mathbf{x} \mid \mathbf{z})$.

The focus of this chapter is mainly on using a quadrature rule to accurately infer Bayesian predictions, but various other statistical quantities that are not considered in this chapter can be inferred straightforwardly without additional model evaluations. For example, moments of the posterior distribution can be determined by integrating suitably chosen polynomials or an empirical cumulative distribution function can be constructed by considering the discrete probability density function $p(\mathbf{x}_k) = w_k$ for all k (which is well defined because all weights are non-negative).

7.3.1. Constructing an adaptive quadrature rule

The proposed procedure consists of iteratively refining a pair consisting of a quadrature rule and a distribution. A typical iteration consists of two steps. Firstly, a quadrature

rule based on the current estimate of the distribution ρ is constructed. Secondly the estimate of the distribution is refined using all available model evaluations of ρ. The quadrature rules are constructed such that a sequence of nested rules is obtained. These steps are illustrated in Figure 7.1.

More specifically, at the start of the i-th iteration a distribution ρ_N is given, accompanied by evaluations of the model u at nodes $\mathbf{x}_0, \ldots, \mathbf{x}_N$. Therefore also evaluations of $\rho(\mathbf{x}_k)$ (for $k = 0, \ldots, N$) are given, since $\rho(\mathbf{x}) = G(u(\mathbf{x}))$ with G explicitly known. The first step consists of constructing a new quadrature rule with $N + M$ nodes that incorporates ρ_N and reuses the available model evaluations, i.e. it is nested. Hence the goal is to determine $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M}$ and $w_0, \ldots, w_{N+M}$ with M as small as possible such that

$$\sum_{k=0}^{N+M} \varphi_j(\mathbf{x}_k)\, w_k = \int_\Omega \varphi_j(\mathbf{x})\, \rho_N(\mathbf{x})\, \mathrm{d}\mathbf{x}, \text{ for all } j = 0, \ldots, D_i.$$

Here, D_i is a free parameter which defines the exactness of the rule, i.e. the dimension of Φ_{D_i}. The second step of the iteration consists of constructing a refined approximation of the distribution ρ using all available evaluations of the model u and distribution ρ (including the M nodes that have been added this iteration). In this thesis, this is done by nearest neighbor interpolation. Consequently ρ_{N+M} is obtained, which is used for the next iteration. The procedure can be initialized using a quadrature rule consisting of a single random node and a proposal distribution $\rho_0(\mathbf{x}) \equiv q(\mathbf{x})$, i.e. by using the prior. The convergence can be assessed by comparing the current quadrature rule estimate with that of a previous iteration.

The only free parameter in this construction is D_i, which can be chosen heuristically. For convergence, it is essential to ensure that $D_i \to \infty$ for $i \to \infty$. In this chapter, two variants are considered: linear growth, i.e. $D_i = \mathcal{O}(i)$, and exponential growth, i.e. $D_i = \mathcal{O}(2^i)$.

The distribution ρ_N is constructed by interpolation of all available point evaluations of ρ. It can be interpreted as a *proposal distribution*, as often used in importance sampling strategies. The efficiency of the approach can be demonstrated mathematically by reusing techniques from these sampling strategies.

The construction of a quadrature rule that incorporates the distribution ρ_N is non-trivial. We will employ the implicit quadrature rule as introduced in Chapter 4 for this purpose, though we emphasize that any methodology to construct quadrature rules for a given distribution can be used in our approach. The construction of this rule has been discussed extensively in Chapter 4. Therefore only most relevant properties that are used specifically in this chapter are discussed in Section 7.3.2. Given the quadrature rule nodes, the distribution that is used to refine the quadrature rule is constructed using nearest neighbor interpolation, which is explained in Section 7.3.3.

7.3.2. The implicit quadrature rule

The implicit quadrature rule is a quadrature rule constructed using samples from a distribution. Hence let $K + 1$ samples $\mathbf{y}_0, \ldots, \mathbf{y}_K$ be given, denoted as the set Y_K. In the iterative procedure considered in this chapter (i.e. Figure 7.1), these are a large number of samples from ρ_N. Moreover, some model evaluations have been performed in previous iterations, so let $X_N \subset \Omega$ be $N + 1$ nodes in Ω where the value of u (and therefore ρ) is

known. These nodes are denoted with $\mathbf{x}_0, \ldots, \mathbf{x}_N$. Notice that ρ_N is typically obtained by nearest neighbor interpolation of these nodes, which we will discuss in more detail in Section 7.3.3.

Then, by applying Algorithm 4.3 for given D_i (see page 88), a subset of M samples from Y_K is obtained, say $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M}$, such that

$$\mathcal{A}_{N+M}^{\rho_N} \varphi_j = \sum_{k=0}^{N+M} \varphi_j(\mathbf{x}_k) w_k = \frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(\mathbf{y}_k) = \mathcal{I}_K^{\rho_N} \varphi_j, \text{ for } j = 0, \ldots, D_i.$$

Here, the notation $\mathcal{A}_{N+M}^{\rho_N}$ is used to denote an implicit quadrature rule operator constructed with samples from ρ_N. The number of samples used for the construction of the quadrature rule is omitted from the notation used in this chapter, since it is not necessarily the case that this rule is an implicit quadrature rule (though that is the case in all examples discussed in this chapter). The operator $\mathcal{I}_K^{\rho_N}$ denotes the sampling integral operator (similarly to (4.1) on page 70) using $K+1$ samples from ρ_N.

The rule is constructed such that the weights w_k are non-negative, i.e. $w_k \geq 0$ (for all k). This ensures that the quadrature rule error is bounded, which can be seen by applying the Lebesgue inequality from (7.5).

Notice that only samples from ρ_N are necessary to construct this quadrature rule, from which the approximate moments are computed. This is an advantage, since in higher dimensional spaces it is often computationally costly to exactly determine the moments of ρ_N.

The nodes $\mathbf{x}_0, \ldots, \mathbf{x}_N$ do generally not form a subset of the samples Y_K. Ideally we would like to have that $M = 0$, which would imply that no new costly model evaluations are required, but this is often not possible as D_i increases with each iteration i. However, the following bound holds, which is a straightforward consequence of (4.10) (see page 78):

$$0 \leq M \leq D_i + 1,$$

such that at most $D_i + 1$ nodes are added to the quadrature rule, consequently obtaining a rule of at most $N + D_i + 2$ nodes.

7.3.3. Construction of the proposal distribution

By using the quadrature rule constructed using the techniques from the previous section, a Bayesian prediction can be inferred by evaluating the model at the quadrature rule nodes and assessing the accuracy using the nesting of the quadrature rules. By using Bayes' rule, evaluations of the (unscaled) posterior are readily available, that is $\rho(\mathbf{x}_k)$ for $k = 0, \ldots, N+M$. The goal is to use these evaluations to construct an approximation of the posterior.

Constructing an approximation of a distribution is non-trivial and there are some specific requirements on the interpolation algorithm used in this chapter. The goal of this section is to outline these requirements, which warrant the algorithm used in this chapter: nearest neighbor interpolation. Various alternative approaches, which have varying advantages and disadvantages, are also discussed.

Prerequisites of the interpolant

The goal is to construct an approximate posterior ρ_{N+M} such that the available $N+M+1$ point evaluations of ρ are taken into account. These point evaluations of the posterior are exact, so in this chapter we construct ρ_{N+M} such that it is interpolatory, i.e.

$$\rho_{N+M}(\mathbf{x}_k) = \rho(\mathbf{x}_k), \text{ for all } k = 0, \ldots, N+M.$$

Obviously, the interpolant should be constructed such that $\rho_{N+M}(\mathbf{x}) \approx \rho(\mathbf{x})$ for all $\mathbf{x} \neq \mathbf{x}_k$ (for any k). It is not necessary that ρ_{N+M} is interpolatory for the construction of the quadrature rule, but it is assumed to be the case in the analysis discussed in Section 7.4.

It is known that ρ is a probability density function, possibly up to a finite constant (the evidence). Therefore the interpolant should be such that $\rho_{N+M}(\mathbf{x}) \geq 0$ for all $\mathbf{x} \in \Omega$. Moreover, since a quadrature rule is constructed using this distribution, all moments must be finite, i.e.

$$\int_\Omega \varphi(\mathbf{x})\, \rho_{N+M}(\mathbf{x})\, \mathrm{d}\mathbf{x} < \infty, \text{ for all polynomials } \varphi.$$

These two conditions ensure that ρ_{N+M} is again a probability density function (possibly up to a constant).

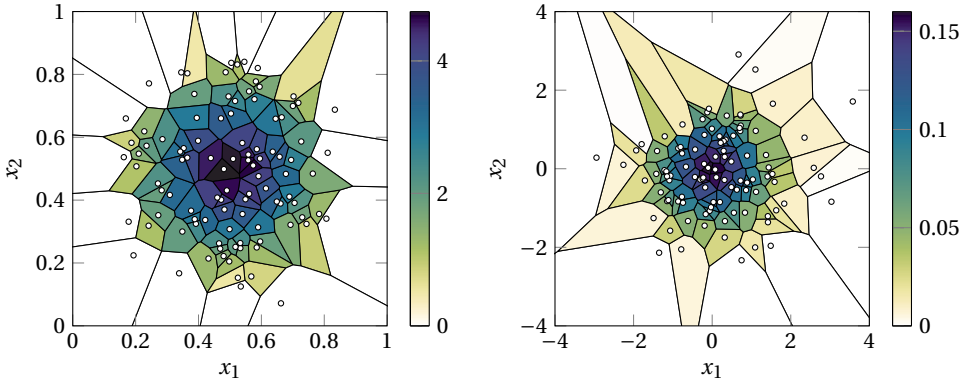
In general it is *not* preferable that the moments of ρ_{N+M} are equal to the moments of the quadrature rule, i.e. in general it should hold that

$$\begin{aligned}
\mathcal{I}^{\rho_{N+M}} \varphi_j &= \int_\Omega \varphi_j(\mathbf{x})\, \rho_{N+M}(\mathbf{x})\, \mathrm{d}\mathbf{x} \\
&\neq \int_\Omega \varphi_j(\mathbf{x})\, \rho_N(\mathbf{x})\, \mathrm{d}\mathbf{x} \\
&= \sum_{k=0}^{N+M} \varphi_j(\mathbf{x}_k)\, w_k \\
&= \mathcal{A}_{N+M}^{\rho_N} \varphi_j,
\end{aligned}$$

for $j = 0, \ldots, D_i$. Recall that $\mathcal{I}^{\rho_{N+M}}$ and $\mathcal{A}_{N+M}^{\rho_N}$ are used to denote an integral weighted with ρ_{N+M} and a quadrature rule weighted with ρ_N respectively, i.e.

$$\begin{aligned}
\mathcal{I}^{\rho_{N+M}} u &= \int_\Omega u(\mathbf{x})\, \rho_{N+M}(\mathbf{x})\, \mathrm{d}\mathbf{x}, \\
\mathcal{A}_{N+M}^{\rho_N} u &= \sum_{k=0}^{N+M} w_k\, u(\mathbf{x}_k).
\end{aligned}$$

The moments estimated by the quadrature rule are generally not equal to the exact moments of the true distribution ρ. Hence forcing that the moments of ρ_{N+M} are equal to the moments as computed using the (current) quadrature rule does not lead to a converging sequence of distributions. In such a case all moments would be equal to the approximation of the quadrature rule that is used to initialize the algorithm (see Figure 7.1).



(a) Beta(4, 4) **(b)** $\mathcal{N}(0,1)$

Figure 7.2: Nearest neighbor interpolations of random samples based on two distributions. In both cases, x_1 and x_2 are identically and independently distributed.

Nearest neighbor interpolation

In this chapter, ρ_{N+M} is constructed by firstly interpolating the likelihood using nearest neighbor interpolating and secondly multiplying the obtained interpolant with the prior. Hence for any $\mathbf{x} \in \Omega$ the interpolant $\rho_{N+M}(\mathbf{x})$ is defined as follows:

$$\rho_{N+M}(\mathbf{x}) = q(\mathbf{z} \mid \mathbf{x}_k)\, q(\mathbf{x}), \text{ with } \mathbf{x}_k = \underset{\mathbf{x}_j \in X_{N+M}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{x}_j\|_2 \text{ and } X_{N+M} = \{\mathbf{x}_0, \ldots, \mathbf{x}_{N+M}\}.$$

The advantages of this approach are among others that the obtained interpolant is always non-negative and globally bounded, i.e. for all $\mathbf{x}$ it holds that

$$0 \leq \rho_{N+M}(\mathbf{x}) \leq \left(\sup_{\mathbf{x}\in\Omega} q(\mathbf{z} \mid \mathbf{x})\right)\left(\sup_{\mathbf{x}\in\Omega} q(\mathbf{x})\right).$$

Moreover, for the purposes of this chapter it can be implemented very efficiently, as the number of nodes used for interpolation is relatively small and the exact structure of the interpolant is not of importance: sampling from ρ_{N+M} can simply be done by means of acceptance rejection sampling using the prior as proposal distribution (which requires a proper prior).

In particular, for the nearest neighbor interpolant it is not necessary to construct the Voronoi tessellation of the nodes, but only to implement a routine that can find the closest node for any given input sample. This can be done very fast and efficiently for large numbers of samples.

Nearest neighbor interpolation is being used often in statistics [9] (it is mostly applied to classification problems). It belongs to the broader class of scattered data interpolation methods [61] and is arguably one of the most straightforward methods to use in this regard. Moreover, it is a local interpolation method, whereas the quadrature rule is a global integration method, which naturally balances exploitation of locality with exploration of the space without any free parameters that have to be tuned.

As mentioned briefly before, a nearest neighbor interpolant can be used directly to construct quadrature rules [29, 48, 77, 114]. Many of these approaches focus on explicitly constructing the nodal set such that these form the centroids (centers of mass) of the cells of the Voronoi tessellation. Notice that it is non-trivial to extend these approaches to the setting of this chapter, since it is not viable to assume that the distribution is computationally tractable. Moreover, it is usually necessary to explicitly construct the Voronoi tessellation to determine the centroids, which significantly hinders the applicability to higher dimensional spaces.

The idea of using a nearest neighbor interpolant to construct a surrogate (i.e. not necessarily a quadrature rule) has also been explored successfully before and the proposed improvements [49, 155] can possibly be considered to enhance the accuracy of the interpolant as considered in this chapter. Often the advantage of exploiting locality is carried over. However, existing approaches usually focus on constructing a surrogate of the computational model, whereas the focus in this chapter is on constructing an interpolant of a distribution. As discussed before, this adds several non-trivial constraints to the approach, which are typically not incorporated in existing methods.

Nearest neighbor interpolation has been illustrated in Figure 7.1 (see page 163). Two other examples of interpolated probability density functions are depicted in Figure 7.2. For sake of illustration, the interpolation nodes of Figure 7.2 are random samples.

Alternative distribution approximation methods

It is not trivial to consider more accurate interpolation methods in this framework, as ensuring that the obtained interpolant is a PDF is difficult. For example, many global methods such as Lagrange interpolation and Gaussian process regression do not yield an interpolant that is non-negative. Extensions to enforce positivity exist, for example for Lagrange interpolation [63, 76]. Often these approaches are only applicable to univariate cases. Gaussian process regression can possibly be used by using a strictly positive prior that is not improper and enforcing finiteness of integrals over the obtained approximation [152], but this is rather involved and conflicts with the convenient Bayesian framework that Gaussian processes are based upon.

Another well-known method to construct a suitable approximation is the kernel density estimate, or more generally, the family of methods based on radial basis functions [27]. However, in the cases in this chapter explicit values of the probability density function are known, which cannot be incorporated in a straightforward fashion in these methods: enforcing specific values of a function in a radial basis function setting does not necessarily yield a positive interpolant.

A promising method that balances the accuracy of polynomial interpolation and piecewise behavior of nearest neighbor interpolation is the Simplex Stochastic collocation method [53, 196, 197, 198], where piecewise high order polynomial interpolation is combined with a Delaunay triangulation of the nodal set. This method converges exponentially for sufficiently smooth functions and is non-oscillatory (i.e. it is bounded from below and above by the minimum and maximum of the model), so it is very suitable for the interpolation of a PDF. However, it requires the explicit construction of the interpolant with the underlying Delaunay triangulation, which severely limits its applicability to multi-dimensional spaces.

Concluding, it is not trivial to consider a more efficient alternative approach to nearest neighbor interpolation that yields a positive interpolant with similar efficiency as nearest neighbor interpolation. It is out of the scope of this thesis to derive such a new interpolation approach. Moreover, nearest neighbor interpolation is sufficiently accurate for the purposes of this chapter, as demonstrated by the numerical examples in Section 7.5.

7.4. Error analysis

The accuracy of the proposed method is studied by assessing the convergence of the sequence of quadrature rules. The overall error we are interested in is the integration error, defined for a given continuous function $f\colon \Omega \to \mathbb{R}$ as follows:

$$e_N(f) := |\mathcal{I}^\rho f - \mathcal{A}_N^{\rho_N} f|.$$

Here the operator $\mathcal{A}_N^{\rho_N}$ has been constructed using the methods outlined in Section 7.3, i.e. for known D_i we have

$$\mathcal{A}_N^{\rho_N} \varphi_j = \sum_{k=0}^{N} \varphi_j(\mathbf{x}_k) w_k = \int_\Omega \varphi_j(\mathbf{x}) \rho_N(\mathbf{x}) \, \mathrm{d}\mathbf{x} = \mathcal{I}^{\rho_N} \varphi_j, \text{ for } j = 0, \ldots, D_i,$$

where we neglect the effect of using samples for the construction of the quadrature rule.

It can be demonstrated that $e_N \to 0$ if $N \to \infty$, provided that f is sufficiently smooth and $D_i \to \infty$ for $i \to \infty$. The details of this are discussed in Section 7.4.1.

A central question is whether this error decays faster than the integration error of a conventional quadrature rule constructed using the prior. As constructing such a quadrature rule does not require interpolation or sampling, there are significantly fewer sources of error. This topic can be addressed by using principles from importance sampling, which is outlined in Section 7.4.2.

The error analysis based on importance sampling neglects any error that occurs due to the usage of samples of ρ_{N+M} instead of the full distribution. As discussed before in this thesis, this is not considered to be a severe limitation, since sampling from ρ_{N+M} does not require costly model evaluations and therefore many samples can be used to minimize this error. However, in this chapter the implicit quadrature rule is used repeatedly and computed quadrature rules are used in subsequent iterations. Therefore the sampling error might be of importance, so it will be briefly considered in Section 7.4.3.

7.4.1. Decay of the integration error

Notice that the integration error $e_N(f)$ can be decomposed into three components: an interpolation error, a sampling error, and a quadrature rule error:

$$\begin{aligned} e_N(f) = |\mathcal{I}^\rho f - \mathcal{A}_N^{\rho_N} f| &= |\mathcal{I}^\rho f - \mathcal{I}^{\rho_N} f + \mathcal{I}^{\rho_N} f - \mathcal{A}_N^{\rho_N} f| \\ &\leq |\mathcal{I}^\rho f - \mathcal{I}^{\rho_N} f| + |\mathcal{I}^{\rho_N} f - \mathcal{A}_N^{\rho_N} f| \\ &= |\mathcal{I}^\rho f - \mathcal{I}^{\rho_N} f| + |\mathcal{I}^{\rho_N} f - \mathcal{I}_K^{\rho_N} f + \mathcal{I}_K^{\rho_N} f - \mathcal{A}_N^{\rho_N} f| \end{aligned}$$

$$\leq \underbrace{|\mathcal{I}^{\rho} f - \mathcal{I}^{\rho_N} f|}_{\text{Interpolation error } I_N} + \underbrace{|\mathcal{I}^{\rho_N} f - \mathcal{I}_K^{\rho_N} f|}_{\text{Sampling error } S_K} + \underbrace{|\mathcal{I}_K^{\rho_N} f - \mathcal{A}_N^{\rho_N} f|}_{\text{Quadrature error } Q_N}$$
$$= I_N + S_K + Q_N. \tag{7.6}$$

Notice that this error decomposition is similar to the decomposition done in Chapter 4 (see (4.6) on page 72), where it was used to illustrate the independence of the quadrature rule error (which depends on N and the smoothness of u) and the sampling error (which depends on K and u to a much lesser extent).

The quadrature rule error Q_N can be bounded using the Lebesgue inequality (see for example (2.12) on page 16):

$$Q_N \leq \left(1 + \sum_{k=0}^{N} |w_k|\right) \inf_{\varphi \in \Phi_{D_i}} \|f - \varphi\|_\infty.$$

Since all quadrature rules constructed in this chapter have positive weights, this error decays if $D_i \to \infty$ (with $i \to \infty$) for sufficiently smooth f.

The interpolation error I_N depends on the specific procedure used to construct the interpolant. It holds that $I_N \to 0$ for $N \to \infty$ if $\|\rho_N - \rho\|_\infty \to 0$. The latter can be demonstrated straightforwardly for nearest neighbor interpolants by demonstrating that the quadrature rule nodes distribute across the space, i.e. the mutual distance between two closest nodes decays to zero. A nearest neighbor interpolant yields a converging interpolant in that case.

To demonstrate that quadrature rule nodes distribute across the space, let $\mathbf{x}_0, \ldots, \mathbf{x}_k, \ldots$ be a sequence of nodes such that $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$ form the nodes of a quadrature rule with positive weights with respect to a fixed distribution ρ for all N. Moreover assume that $\mathbf{x}_k \in \Omega$ for all $k \in \mathbb{N}$. Suppose $S \subset \Omega$ is such that there is no $k \in \mathbb{N}$ such that $\mathbf{x}_k \in S$. To demonstrate that the nodes distribute across the space, we will show that S as considered here is a null set, i.e.

$$\int_S \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} = 0,$$

which implies that the mutual distance between the quadrature rule nodes vanishes. The proof is indirect, so let $u_S \in C^\infty(\Omega)$ be a smooth function such that $u_S(\mathbf{x}) = 0$ for all $\mathbf{x} \notin S$ and such that

$$\int_\Omega u_S(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} > 0.$$

Such a u_S only exists if S is not a null set. A univariate example of such a function is the bump function, e.g. if $S = [-1, 1]$ then

$$u_S(x) = \begin{cases} \exp\left(-\frac{1}{1-x^2}\right) & \text{if } |x| < 1, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, multivariate bump functions can be constructed by products of univariate bump functions defined on a hypercube in S. Due to the smoothness of u_S, there exists a sequence of polynomials φ_N such that $\|u_S - \varphi_N\|_\infty \to 0$ for $N \to \infty$. Hence *any*

quadrature rule operator $\mathcal{A}_N^\rho$ with positive weights yields a converging estimate of $\mathcal{I}^\rho u_S$, i.e.

$$\mathcal{A}_N^\rho u_S = \sum_{k=0}^{N} u_S(\mathbf{x}_k)\, w_k \to \int_\Omega u_S(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x}, \text{ for } N \to \infty.$$

The sequence of nodes is such that there is no $\mathbf{x}_k \in \Omega$ for all $k \in \mathbb{N}$. Since the quadrature rule converges, this yields that the integral of u_S must vanish:

$$\int_\Omega u_S(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_S u_S(\mathbf{x})\, \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} = 0.$$

This holds for any smooth function defined in this way, which is only possible if

$$\int_S \rho(\mathbf{x})\, \mathrm{d}\mathbf{x} = 0.$$

Notice that positivity of the weights is essential here, as otherwise it is not guaranteed that the quadrature rule approximation of any smooth function converges.

A similar derivation can be formulated for the nodes deduced with the methodology proposed in this chapter, although it is essential that the interpolant of ρ is globally non-negative. Otherwise the function u_S could be constructed such that it is positive if integrated with respect to ρ, but would yield an integral equal to zero if integrated with respect to ρ_N.

The sampling error S_K is independent of N, and can therefore be reduced without any additional costly model evaluations by considering more samples from ρ. The exact convergence rate depends on the constructed sequence of samples, which will be further discussed in Section 7.4.3.

Concluding, if $D_i \to \infty$ and $N \to \infty$, it holds that $e_N(f) \to S_K$ provided that f can be approximated using polynomials, which is the case if

$$\inf_{\varphi \in \Phi_{D_i}} \|\varphi - f\|_\infty \to 0.$$

Moreover, if also $K \to \infty$, it holds that $e_N(f) \to 0$. This demonstrates the convergence of our method for sufficiently smooth functions, but this does not demonstrate that our method converges faster than constructing a conventional quadrature rule with respect to the prior, since such a rule does not have an interpolation error. This is the main topic of Section 7.4.2.

7.4.2. Importance sampling

The proposed quadrature rule of this chapter has three sources of error: an interpolation error I_N, a sampling error S_K, and a quadrature rule error Q_N. A quadrature rule constructed using the prior (e.g. by means of a sparse grid) only has the quadrature rule error, i.e. (7.6) simplifies to $e_N(f) = Q_N$.

The main difference is that a rule that only uses the prior requires a different integrand, i.e. $\mathcal{I}^\rho f$ is approximated by means of $\mathcal{A}_N^{q(.)}(q(\mathbf{z}\,|\,\mathbf{x}) f(\mathbf{x}))$, whereas the quadrature rule proposed in this chapter uses $f(\mathbf{x})$ as integrand. In this section we derive in which cases the proposed quadrature rule needs fewer model evaluations than a conventional

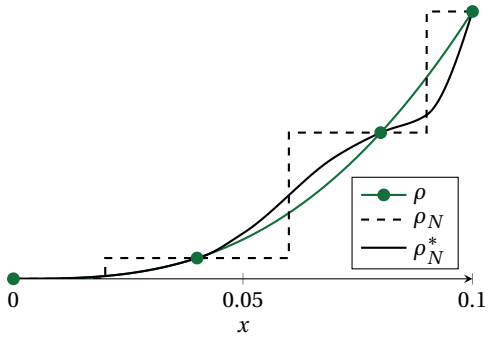


Figure 7.3: A sketch of a possible interpretation of ρ_N^*. In this particular case, $\rho(x)/\rho_N(x)$ is unbounded for $x \to 0$. Therefore, a ρ_N^* is introduced that yields a bounded fraction $\rho(x)/\rho_N^*(x)$ for $x \approx 0$. Corrections elsewhere in the domain are necessary to enforce that the raw moments of ρ_N^* are equal to those of ρ_N.

quadrature rule to reach a similar accuracy. Without loss of generality, we assume that both ρ and ρ_N are probability density functions, which yields that $\|\mathcal{I}^\rho\|_\infty = \|\mathcal{I}^{\rho_N}\|_\infty = 1$.

The analysis is based on the main principle of importance sampling, a technique that is used to estimate integrals with respect to a target distribution if samples from a proposal distribution are known. Even though the main interest is not in drawing samples from the posterior, the following key idea of importance sampling can be readily used: let $\psi(\mathbf{x})$, the so-called proposal distribution, be a probability density function with the same support as ρ. Then

$$\mathcal{I}^\rho f = \int_\Omega f(\mathbf{x})\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_\Omega f(\mathbf{x})\,\frac{\rho(\mathbf{x})}{\psi(\mathbf{x})}\,\psi(\mathbf{x})\,\mathrm{d}\mathbf{x} = \mathcal{I}^\psi\left(f\,\frac{\rho}{\psi}\right).$$

Hence it is not necessary to calculate a possibly expensive integral with respect to the distribution ρ, but to calculate one with respect to the proposal distribution ψ.

It is a natural idea to use the proposal distribution $\psi = \rho_N$ for the analysis, but this can severely limit the applicability of the Lebesgue inequality from (7.5), as $\|\rho/\rho_N\|_\infty$ can be large even if $\|\rho - \rho_N\|_\infty$ is small. Nonetheless, in many cases ρ/ρ_N can be globally bounded. For example, if a uniform prior and a Gaussian likelihood is considered, ρ_N can be used as proposal distribution. However, an example where this is not the case is if ρ is a Beta distribution and a node is placed on the boundary of Ω. In that case ρ/ρ_N is unbounded on the boundary.

To alleviate this, we choose a slightly different approach, and use a proposal distribution ρ_N^* such that it is interpolatory and integrates the same moments as ρ_N, i.e.

$$\rho_N^*(\mathbf{x}_k) = \rho_N(\mathbf{x}_k) = \rho(\mathbf{x}_k), \text{ for } k = 0, \ldots, N, \tag{7.7}$$

and

$$\mathcal{I}^{\rho_N^*}\varphi_j = \mathcal{I}^{\rho_N}\varphi_j, \text{ for } j = 0, \ldots, D_i. \tag{7.8}$$

These two conditions define a set of possible proposal distributions, which we will call R in this section, i.e. $\rho_N^* \in R$ if it solves (7.7) and (7.8). See Figure 7.3 for a sketch how

such a ρ_N^* can be interpreted. The exact shape of ρ_N^* used in this particular example is not relevant for the analysis (e.g. ρ_N^* can also be discontinuous). The only conditions on ρ_N^* are (7.7), (7.8), and $\|\rho/\rho_N^*\|_\infty < \infty$.

It is not necessary to construct a ρ_N^* to use the method proposed in this work, since only samples from ρ_N are used. The distribution ρ_N^* is only a tool in the analysis discussed in this section. Notice that $\rho_N \in R$, so R is a well-defined non-empty set. However, as explained above it is not desirable to use $\rho_N^* = \rho_N$ in the subsequent analysis as that might lead to a potentially large $\|\rho/\rho_N^*\|_\infty$.

Conditions (7.7) and (7.8) ensure firstly that the moments of ρ_N are equal to the moments of ρ_N^*. Secondly, integrating ρ_N, ρ, or ρ_N^* using a quadrature rule with nodes $\mathbf{x}_0, \dots, \mathbf{x}_N$ yields the same result. The latter is especially useful in importance sampling, since this implies that for all $k = 0, \dots, N$:

$$1 = \frac{\rho(\mathbf{x}_k)}{\rho_N(\mathbf{x}_k)} = \frac{\rho(\mathbf{x}_k)}{\rho_N^*(\mathbf{x}_k)}.$$

This property will be used extensively in the remainder of this section.

To this end, fix an arbitrary $\rho_N^* \in R$, i.e. one that satisfies (7.7) and (7.8), as proposal distribution. Then

$$\mathcal{I}^\rho f = \mathcal{I}^{\rho_N}\left(f\frac{\rho}{\rho_N}\right) = \mathcal{I}^{\rho_N^*}\left(f\frac{\rho}{\rho_N^*}\right). \tag{7.9}$$

A similar argument can be applied to rewrite $\mathcal{A}_N^{\rho_N}$:

$$\mathcal{A}_N^{\rho_N} f = \sum_{k=0}^{N} f(\mathbf{x}_k)\, w_k = \sum_{k=0}^{N} \frac{\rho(\mathbf{x}_k)}{\rho_N(\mathbf{x}_k)} f(\mathbf{x}_k)\, w_k = \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N}\right),$$

where it is being used that $\rho_N(\mathbf{x}_k) = \rho(\mathbf{x}_k)$ for all $k = 0, \dots, N$. Hence the following is obtained:

$$\mathcal{A}_N^{\rho_N} f = \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N}\right) = \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N^*}\right). \tag{7.10}$$

These expressions, i.e. the defining properties of ρ_N^* as described by (7.7) and (7.8), and the derived relations for the integral and quadrature rule operator, as described by (7.9) and (7.10), form sufficient ingredients to bound $e_N(f)$. The derivation is similar to the derivation of the Lebesgue inequality (recall (7.5)), but some bookkeeping is required to keep track of the correct proposal distribution:

$$\begin{aligned} e_N(f) &= |\mathcal{I}^\rho f - \mathcal{A}_N^{\rho_N} f| \\ &= \left|\mathcal{I}^\rho f - \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N^*}\right)\right| \\ &= \left|\mathcal{I}^\rho f - \mathcal{I}^{\rho_N^*}\varphi + \mathcal{I}^{\rho_N^*}\varphi - \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N^*}\right)\right|. \end{aligned}$$

Here, $\varphi \in \Phi_{D_i}$. Hence the quadrature rule exactly integrates φ, i.e. $\mathcal{A}_N^{\rho_N}\varphi = \mathcal{I}^{\rho_N}\varphi$, obtaining the following:

$$e_N(f) = \left|\mathcal{I}^{\rho_N^*}\left(f\frac{\rho}{\rho_N^*}\right) - \mathcal{I}^{\rho_N^*}\varphi + \mathcal{A}_N^{\rho_N}\varphi - \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N^*}\right)\right|,$$

where we use that $\mathcal{I}^{\rho_N^*}\varphi = \mathcal{I}^{\rho_N}\varphi = \mathcal{A}_N^{\rho_N}\varphi$ for $\varphi \in \Phi_{D_i}$, which follows from (7.8). Concluding, the following inequality is obtained:

$$\begin{aligned} e_N(f) &\leq \left|\mathcal{I}^{\rho_N^*}\left(f\frac{\rho}{\rho_N^*}\right) - \mathcal{I}^{\rho_N^*}\varphi\right| + \left|\mathcal{A}_N^{\rho_N}\varphi - \mathcal{A}_N^{\rho_N}\left(f\frac{\rho}{\rho_N^*}\right)\right| \\ &\leq \left(\|\mathcal{I}^{\rho_N^*}\|_\infty + \|\mathcal{A}_N^{\rho_N}\|_\infty\right)\left\|f\frac{\rho}{\rho_N^*} - \varphi\right\|_\infty \\ &= 2\left\|f\frac{\rho}{\rho_N^*} - \varphi\right\|_\infty. \end{aligned} \tag{7.11}$$

Notice that in a Bayesian framework the ratio ρ/ρ_N^* can be expressed independently of the prior as follows:

$$\frac{\rho(\mathbf{x})}{\rho_N^*(\mathbf{x})} = \frac{q(\mathbf{z}\,|\,\mathbf{x})\,q(\mathbf{x})}{q_N^*(\mathbf{z}\,|\,\mathbf{x})\,q(\mathbf{x})} = \frac{q(\mathbf{z}\,|\,\mathbf{x})}{q_N^*(\mathbf{z}\,|\,\mathbf{x})}.$$

Here, with a little abuse of notation, $q_N^*(\mathbf{z}\,|\,\mathbf{x})$ represents the nearest neighbor interpolant of the likelihood. Similarly to ρ_N^*, it is not necessary to know any details of $q_N^*(\mathbf{z}\,|\,\mathbf{x})$, since it is only used here to demonstrate independence from the prior.

By choosing $\rho_N^* \in R$ and $\varphi \in \Phi_{D_i}$ such that the obtained norm from (7.11) is minimal, an inequality similar to the Lebesgue inequality from (7.5) is obtained:

$$e_N(f) = |\mathcal{I}^\rho f - \mathcal{A}_N^{\rho_N} f| \leq 2\inf_{\rho_N^*}\inf_{\varphi\in\Phi_{D_i}}\left\|f\frac{\rho}{\rho_N^*} - \varphi\right\|_\infty = 2\inf_{q_N^*}\inf_{\varphi\in\Phi_{D_i}}\left\|f\frac{q(\mathbf{z}\,|\,\cdot)}{q_N^*(\mathbf{z}\,|\,\cdot)} - \varphi\right\|_\infty. \tag{7.12}$$

Here, $\rho_N^* = q_N^*(\mathbf{z}\,|\,\cdot)\,q(\cdot) \in R$ is according to (7.7) and (7.8).

It is instructive to compare the obtained inequality with that of a conventional quadrature rule constructed with respect to the prior. If such a quadrature rule, say $\mathcal{A}_N^{q(\cdot)}$, is given, then by using the Lebesgue inequality its error can be bounded as follows (notice the difference with (7.11)):

$$\left|\mathcal{I}^\rho f - \mathcal{A}_N^{q(\cdot)}\left(f\,q(\mathbf{z}\,|\,\cdot)\right)\right| = \left|\mathcal{I}^{q(\cdot)}\left(f\,q(\mathbf{z}\,|\,\cdot)\right) - \mathcal{A}_N^{q(\cdot)}\left(f\,q(\mathbf{z}\,|\,\cdot)\right)\right| \leq 2\inf_{\varphi\in\Phi_{D_i}}\|f\,q(\mathbf{z}\,|\,\cdot) - \varphi\|_\infty, \tag{7.13}$$

where we use that the prior is not improper (as otherwise $\|\mathcal{I}^{q(\cdot)}\|_\infty \neq 1$).

It is not generally true that the right-hand side of (7.12), i.e. the error of our proposed adaptive rule, is smaller than the right-hand side of (7.13), i.e. the error of conventional rules with respect to the prior. For example, the latter is smaller if both f and $q(\mathbf{z}\,|\,\cdot)$ are polynomials (which is rarely the case in Bayesian model calibration). In general, it is the case that a quadrature rule constructed using the iterative procedure of this chapter outperforms a quadrature rule constructed using the prior if f can be approximated much more efficiently by polynomials than $f\,q(\mathbf{z}\,|\,\cdot)$. To see this, we assume that the approximation of f converges *strictly* faster than the approximation of $f\,q(\mathbf{z}\,|\,\cdot)$, so using little-o notation:

$$\inf_{\varphi\in\Phi_{D_i}}\|f - \varphi\|_\infty = o\left(\inf_{\varphi\in\Phi_{D_i}}\|f\,q(\mathbf{z}\,|\,\cdot) - \varphi\|_\infty\right). \tag{7.14}$$

To compare both quadrature rules, recall that ρ_N is constructed such that $\rho_N \to \rho$. Then there exists a $\rho_N^* \in R$ such that

$$\|\rho/\rho_N^*\|_\infty \to 1, \text{ for } N \to \infty.$$

Hence there exist sequences $\{a_N\}_N$ and $\{b_N\}_N$ such that $a_N \le \|\rho/\rho_N^*\|_\infty \le b_N$ for all N with $a_N \uparrow 1$ and $b_N \downarrow 1$ for $N \to \infty$*. Hence

$$\begin{aligned}\inf_{\varphi\in\Phi_{D_i}} \left\| f\frac{\rho}{\rho_N^*} - \varphi \right\|_\infty &\le \max\left(\inf_{\varphi\in\Phi_{D_i}} \|a_N f - \varphi\|_\infty; \inf_{\varphi\in\Phi_{D_i}} \|b_N f - \varphi\|_\infty \right) \\ &\le \max\left(a_N \inf_{\varphi\in\Phi_{D_i}} \|f - \varphi\|_\infty; b_N \inf_{\varphi\in\Phi_{D_i}} \|f - \varphi\|_\infty \right).\end{aligned}$$

So for all $\varepsilon > 0$ there exists an N_1 such that

$$\inf_{\varphi\in\Phi_{D_i}} \left\| f\frac{\rho}{\rho_N^*} - \varphi \right\|_\infty \le (1+\varepsilon) \inf_{\varphi\in\Phi_{D_i}} \|f - \varphi\|_\infty, \text{ for all } N \ge N_1.$$

Thus, for $N \ge N_1$ the following bound is obtained:

$$e_N(f) \le 2(1+\varepsilon) \inf_{\varphi\in\Phi_{D_i}} \|f - \varphi\|_\infty. \tag{7.15}$$

By combining this bound with (7.14), it follows that (7.12) is sharper than (7.13).

Concluding, if incorporating the likelihood in the integrand (obtaining $fq(\mathbf{z} \mid \cdot)$) yields an integrand that is difficult to approximate using polynomials, the proposed adaptive method has a sharper error bound than conventional rules. This is described by (7.14) and is often the case if the likelihood is informative (i.e. it has small standard deviation) or if the model is irregular (which yields an even more irregular likelihood). For large N the error of the adaptive quadrature rule proposed in this chapter behaves similar to that of a conventional quadrature rule constructed with respect to the *posterior*, as described by (7.15), even though this conventional quadrature rule can usually not be determined without requiring large numbers of evaluations of the posterior.

7.4.3. Sampling error

As an arbitrarily sized sample set can be easily drawn from the distribution ρ_N for any N, the sampling error at least decays with rate $\sqrt{K}$, i.e. for $K \to \infty$, the error converges to a Gaussian distribution as follows:

$$|\mathcal{I}^{\rho_N} f - \mathcal{I}_K^{\rho_N} f| \sim \mathcal{N}(\mathcal{I}^{\rho_N} f, \sigma^2),$$

with $\sigma = \sigma(f)/\sqrt{K}$. Here, $\sigma(f)$ is the standard deviation of f with respect to ρ_N, i.e.

$$\left(\sigma(f)\right)^2 = \mathcal{I}^{\rho_N}\left((f - \mathcal{I}^{\rho_N} f)^2\right).$$

*In other words, a_N converges to 1 from below and b_N converges to 1 from above for $N \to \infty$.

In this chapter, acceptance rejection sampling is used to draw large numbers of samples from ρ_N. Several improvements exist to construct sequences that convergence faster, for example quasi-Monte Carlo sequences [31, 132]. We emphasize that these approaches can be used straightforwardly in the framework explained in this chapter.

The number of samples K can be chosen independently of the number of nodes N under consideration, so the sampling error can be made arbitrarily small without any costly model evaluations. Choosing a larger sample set does however mean that it takes more computational effort to determine the implicit quadrature rule. In practice, the interpolation and quadrature rule error dominate both the error of the quadrature rule and the cost of the procedure (through model evaluations). Only if the model is analytic and the likelihood is not informative, the quadrature rule error and interpolation error decay rapidly enough to make the sampling error dominate.

7.5. Numerical examples

The properties of the method derived in this chapter are illustrated using various numerical examples. Two types of cases are discussed.

Firstly, in Section 7.5.1 three different classes of analytical test cases are discussed to illustrate the key properties of our method. Secondly, the main motivation for this work is to apply Bayesian calibration to the closure coefficients of turbulence models, similarly to the case discussed in Chapter 6. To demonstrate the applicability of our method to this case, we calibrate the closure coefficients of the Spalart–Allmaras one-equation model using measurement data of the flow over a transonic airfoil and infer predictions using the calibrated parameters. This problem is further considered in Section 7.5.2.

7.5.1. Analytical test cases

Firstly, a univariate case is discussed where the sampling and interpolation error as introduced in (7.6) can be observed well. The quadrature error is studied to a lesser extent, since it has already been considered extensively in previous chapters. Secondly, three two-dimensional test functions are calibrated using numerically generated data. The test functions, based on the Genz test functions [66] that have been used before in this thesis, are tuned such that the likelihood is very informative. Finally, all six Genz test functions are calibrated using numerically generated data. The domain of these functions is five-dimensional and a comparison is made with conventional quadrature rules such as the tensor grid and the Smolyak sparse grid.

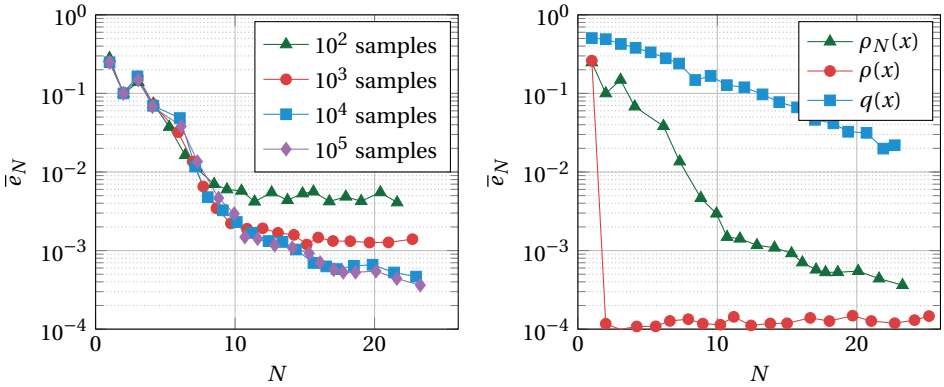
Effect of sampling and interpolation error

Let $\Omega = [0,1]$ and consider a uniform prior in conjunction with a Beta(40,60) distributed likelihood, which yields:

$$\rho(x) \propto x^{40}(1-x)^{60}.$$

To assess the effect of the *sampling* error, the proposed adaptive quadrature rule is constructed using $K = 10^n$ samples with $n = 2,3,4,5$. The obtained quadrature rule is used to approximate the mean of $\rho(x)$:

$$\int_\Omega \varphi_1(x)\,\rho(x)\,\mathrm{d}x = \int_\Omega x\,\rho(x)\,\mathrm{d}x \approx \sum_{k=0}^{N} x_k\,w_k = \sum_{k=0}^{N} \varphi_1(x_k)\,w_k. \tag{7.16}$$



(a) Effect of varying number of samples

(b) Effect of varying distribution

Figure 7.4: Integration error as a function of the number of nodes. In the left figure, the error of an adaptive quadrature rule is depicted for various numbers of samples. In the right figure, the error of an implicit quadrature rule constructed using various distributions is depicted, for fixed $K = 10^5$.

The obtained estimate is compared to the true mean (which is 2/5). The sampling error depends on the randomly generated samples, so the experiment is repeated 50 times and the obtained errors are averaged to illustrate general behavior of the error. The results are gathered in Figure 7.4a, where the averaged error is denoted by $\overline{e}_N$.

For small N, it is clear that the errors of all quadrature rules converge geometrically, which implies that the quadrature error or the interpolation error dominates. For large N, the error does not decay further due to the limited number of samples. The point where the error stops decaying depends on the number of samples used, i.e. a larger number of samples results in a smaller overall error.

To assess the effect of the *interpolation* error, three different implicit quadrature rules are constructed. The first rule is constructed using the adaptive method proposed in this chapter, i.e. using $\rho_N(x)$ like in Figure 7.4a. This quadrature rule has a sampling error and interpolation error, but no quadrature error since we are comparing the first moment (see (7.16)). The second rule is constructed using the *exact* distribution, i.e. $\rho(x)$, which results in a rule that only has a sampling error. The third rule is constructed using the uniform distribution, i.e. using $q(x)$. This rule has a sampling error and a quadrature error, as the likelihood needs to be incorporated in the integrand. All three quadrature rules are used to approximate the mean of the posterior. The rules constructed using $\rho_N(x)$ and $\rho(x)$ can estimate the mean using (7.16), whereas the rule that is constructed using $q(x)$ requires integration of $f(x) = \rho(x) \cdot x$. All rules are constructed using 10^5 samples and again the experiment is repeated 50 times to demonstrate general behavior of the rules. The results are gathered in Figure 7.4b.

The error of the rule constructed using samples from $\rho(x)$ immediately saturates to a level where the sampling error dominates, as no interpolation is used and the quadrature rule error vanishes since the error estimate from (7.16) is based on the mean

(i.e. the first moment). The error of the rule constructed using the prior $q(x)$ converges geometrically, as the PDF of the Beta(40, 60) distribution is a polynomial. A significant improvement is noticed by using our proposed adaptive proposal distribution $\rho_N(x)$. Initially a larger rate of convergence is obtained (up to approximately $N = 10$), as the interpolant provides rapid convergence. For large N the rate of convergence equals the rate of the rule constructed using $q(x)$.

This test case demonstrates that using samples that have been obtained directly from $\rho(x)$ is obviously preferable, but this is intractable in practice. The adaptive proposal distribution has a clear advantage over using a rule that only incorporates the prior.

Two-dimensional Genz test functions

The previously considered test case demonstrates the advantages of using an adaptive quadrature rule. To illustrate the nodal placement of this rule and to further compare this rule to implicit quadrature rules constructed with respect to the prior, the calibration of three two-dimensional Genz test functions [66] with artificially generated data is considered (see Table 4.1 on page 89). Therefore, let $d = 2$, $\mathbf{x}^* = (1/2, 1/2)^{\mathrm{T}}$, and consider the following functions:

$$u_1(\mathbf{x}) = \prod_{i=1}^{d} \left[\frac{1}{4} + \left(x_i - \frac{1}{2} \right)^2 \right]^{-1}, \qquad \text{(Product Peak)}$$

$$u_2(\mathbf{x}) = \exp\left(-\sum_{i=1}^{d} \left| x_i - \frac{1}{2} \right| \right), \qquad (C^0 \text{ function})$$

$$u_3(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 > 3/5 \text{ and } x_2 > 3/5, \\ \exp\left(\sum_{i=1}^{d} x_i\right) & \text{otherwise.} \end{cases} \qquad \text{(Discontinuous function)}$$

These functions are specifically crafted such that they have difficult characteristics for numerical integration routines at or around $\mathbf{x} = \mathbf{x}^*$. The statistical model under consideration is

$$z_k = u_g(\mathbf{x}) + \varepsilon, \text{ with } \varepsilon \sim \mathcal{N}(0, \sigma^2) \text{ and } \sigma = 1/10, \text{ for } k = 1, \ldots, 20. \tag{7.17}$$

Here, u_g (with $g = 1, 2, 3$) denotes one of the functions under consideration and $\mathbf{z} = (z_1, \ldots, z_{20})^{\mathrm{T}}$ are 20 "measurements" obtained by $z_k = u_g(\mathbf{x}^*) + n_k$ with n_k a sample from $\mathcal{N}(0, \sigma^2)$. Hence the data is centered around the defining ("difficult") characteristics of the test functions.

The goal is to infer $\mathbf{x}$ from (7.17), or equivalently, characterize the posterior $q(\mathbf{x} \mid \mathbf{z})$ with a uniform prior and Gaussian likelihood defined in $\Omega = [0, 1]^d$, i.e. we use $\rho(\mathbf{x}) = q(\mathbf{z} \mid \mathbf{x})\, q(\mathbf{x})$ with

$$\begin{aligned} q(\mathbf{x}) &= \begin{cases} 1 & \text{if } \mathbf{x} \in \Omega, \\ 0 & \text{otherwise,} \end{cases} \\ q(\mathbf{z} \mid \mathbf{x}) &\propto \exp\left[-\frac{1}{2} \frac{\sum_{k=1}^{20} (u(\mathbf{x}) - z_k)^2}{\sigma^2} \right]. \end{aligned} \tag{7.18}$$

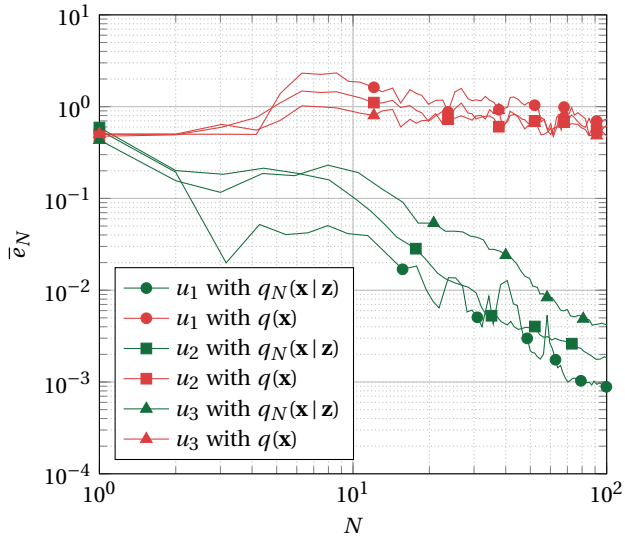
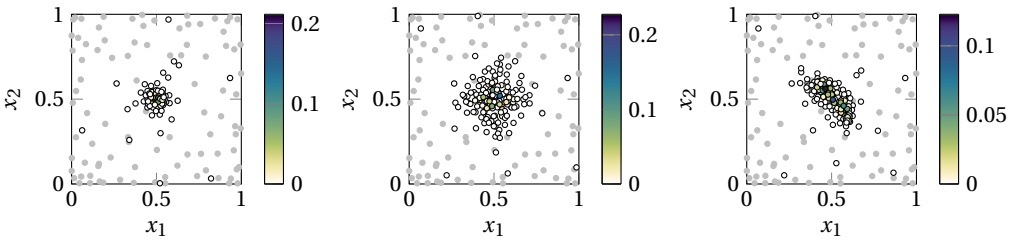


Figure 7.5: Convergence of the mean error determined with the new adaptive implicit quadrature rule or a quadrature rule determined using the prior. Equal colors refer to the same quadrature rule. Equal symbols refer to the same function.



(a) u_1 **(b)** u_2 **(c)** u_3

Figure 7.6: Examples of quadrature rules with $D = 65$ (exact for all polynomials up to degree 10) obtained by the adaptive implicit quadrature rule. The colors indicate the weights of the quadrature rule nodes. The nodes of a quadrature rule of the same polynomial degree, but constructed with respect to the prior, are depicted in gray.

Table 7.1: The multivariate test functions from Genz [66]. All d-variate functions depend on the d-element vectors $\mathbf{a}$ and $\mathbf{b}$. The vector $\mathbf{b}$ is an offset parameter to shift the function. The vector $\mathbf{a}$ describes the degree to which the family attribute is present.

Integrand Family	Attribute
$u_1(\mathbf{x}) = \cos\left(2\pi b_1 + \sum_{i=1}^d a_i x_i\right)$	Oscillatory
$u_2(\mathbf{x}) = \prod_{i=1}^d \left(a_i^{-2} + (x_i - b_i)^2\right)^{-1}$	Product Peak
$u_3(\mathbf{x}) = \left(1 + \sum_{i=1}^d a_i x_i\right)^{-(d+1)}$	Corner Peak
$u_4(\mathbf{x}) = \exp\left(-\sum_{i=1}^d a_i^2 (x_i - b_i)^2\right)$	Gaussian
$u_5(\mathbf{x}) = \exp\left(-\sum_{i=1}^d a_i \lvert x_i - b_i\rvert\right)$	C^0 function
$u_6(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 > b_1 \text{ or } x_2 > b_2 \\ \exp\left(\sum_{i=1}^d a_i x_i\right) & \text{otherwise} \end{cases}$	Discontinuous

The quantity used to measure the error of the quadrature rules is the maximum error of the mean of the posterior, i.e.

$$e_N = \begin{cases} \max\Big(\lvert\mathcal{A}_N\varphi_1 - \mathcal{I}^\rho\varphi_1\rvert, \lvert\mathcal{A}_N\varphi_2 - \mathcal{I}^\rho\varphi_2\rvert\Big) & \text{if } \mathcal{A}_N \text{ w.r.t. } \rho_N(\mathbf{x}), \\ \max\Big(\Big\lvert\mathcal{A}_N\Big(\varphi_1 q(\mathbf{z}\mid x_1)\Big) - \mathcal{I}^\rho\varphi_1\Big\rvert, \Big\lvert\mathcal{A}_N\Big(\varphi_2 q(\mathbf{z}\mid x_2)\Big) - \mathcal{I}^\rho\varphi_2\Big\rvert\Big) & \text{if } \mathcal{A}_N \text{ w.r.t. } q(\mathbf{x}), \end{cases}$$

where $\mathcal{A}_N$ denotes a quadrature rule operator constructed using the adaptive method proposed in this chapter (i.e. with respect to ρ_N) or solely using the prior (i.e. with respect to $q(\mathbf{x})$). In both cases the weights are scaled such that $\sum_k w_k = 1$. All quadrature rules are constructed using a linearly growing D_i, i.e. in iteration i it is enforced that the quadrature rule integrates all $\varphi \in \Phi_i$ exactly (with Φ_i as introduced in Section 7.2.2). Recall that if $\mathbf{x} = (x_1, x_2)$ we have that $\varphi_1(\mathbf{x}) = x_1$ and $\varphi_2(\mathbf{x}) = x_2$.

There is randomness both in the generation of the quadrature rules and in the generation of the measurement data. Therefore, the error e_N is determined 25 times and the obtained errors are averaged. The averaged error is denoted by $\overline{e}_N$. The convergence results are summarized in Figure 7.5. Examples of quadrature rules obtained by applying the proposed procedure are depicted in Figure 7.6. There are still oscillations visible, regardless of the averaging, though the error decay can clearly be observed.

Notice that the adaptive implicit quadrature rule outperforms the conventional implicit quadrature rule in all three cases. The scatter plots of the nodes illustrate that the quadrature rules are indeed tailored specifically to the posterior under consideration. Since all rules have positive weights, estimates determined using these rules are unconditionally stable. This is an aforementioned advantage of the proposed method: the quadrature rules exhibit both exploitation (due to the interpolant) and exploration (due to the spread of the nodes) without suffering from instability.

Five-dimensional Genz test functions

In this section, the calibration of the five-dimensional Genz test functions is considered. The setting is similar to the one of the previous section, i.e. let $d = 5$ and $\mathbf{x}^* \in \mathbb{R}^d$ with $x_k^* = 1/2$. Let u_g be the g-th Genz test function, as defined by Table 4.1 on page 89 (and reproduced as Table 7.1 in this chapter). The vectors $\mathbf{a} = (a_1, \ldots, a_d)^{\mathrm{T}}$ and $\mathbf{b} = (b_1, \ldots, b_d)^{\mathrm{T}}$ are chosen randomly in this test case, as done in previous chapters. Recall that the elements of $\mathbf{a}$ enlarge the defining "difficult" property of the function and $\mathbf{b}$ translates the function in space. The statistical model under consideration is the same model considered in the previous section, see (7.17) and (7.18). Notice that, due to the random nature of $\mathbf{b}$, the probability that the difficult property of the functions is around $\mathbf{x} = \mathbf{x}^*$ is small.

Four quadrature rules are considered to determine the mean of the posterior. Firstly, the proposed quadrature rule with a nearest neighbor interpolant is used. At each iteration the exactness of the quadrature rule is doubled, starting with $D_0 = 2^0$ up to $D_{10} = 2^{10}$. In other words, subsequent quadrature rules integrate larger numbers of polynomials (recall that Φ_D denotes the space of $D+1$ polynomials, sorted graded lexicographically). Secondly, an implicit quadrature rule without interpolation is used, generated using solely the prior. Finally, to compare the results with conventional quadrature rules, a tensor grid and a Smolyak sparse grid generated with a Clenshaw–Curtis quadrature rule are used. A linearly growing sequence of Clenshaw–Curtis quadrature rules is considered, so the tensor grid and the Smolyak sparse grid do not form a sequence of nested quadrature rules. This is done to keep the number of nodes of the multivariate quadrature rules tractable. We do not present the results obtained with Markov chain Monte Carlo, since the number of model evaluations considered here is too small to obtain a "burned-in" sequence.

For each quadrature rule, the experiment is repeated 25 times and the reported errors are averaged. For each experiment, the vectors $\mathbf{a}$ and $\mathbf{b}$ are drawn randomly from the five-dimensional unit hypercube and $\mathbf{a}$ is subsequently scaled such that $\|\mathbf{a}\|_2 = 5/2$. Moreover, for each experiment different samples are used to generate the implicit quadrature rules. To introduce randomness in the tensor grid and Smolyak sparse grid, the "exact" value $\mathbf{x}^*$ is randomized in the domain. This *improves* the results of calibration with the Clenshaw–Curtis rules, as it otherwise would have a node at $\mathbf{x} = \mathbf{x}^*$ and therefore would significantly overestimate the moments. The quantity used to measure the error of the quadrature rules is the same as in the previous test case, i.e. e_N is the maximum error of all first-order raw moments (which are $\varphi_1, \ldots, \varphi_5$) and $\overline{e}_N$ denotes the value of e_N averaged over 25 experiments. The "exact" mean is determined using a Monte Carlo sample from the prior. The number of samples used is the same number used to generate the implicit quadrature rules, such that the sampling error can be observed well (in that case all errors saturate). The obtained results are depicted in Figure 7.7.

The first four test functions are analytic, so for these test functions the quadrature rules perform best. The flattening of the error of the third test function occurs due to the sampling error of the *exact* mean (and not due to the sampling error of the quadrature rule). Moreover, the results of the third test function are best, since the corner peak varies only near a corner (hence the name) and therefore yields an almost uniform posterior.

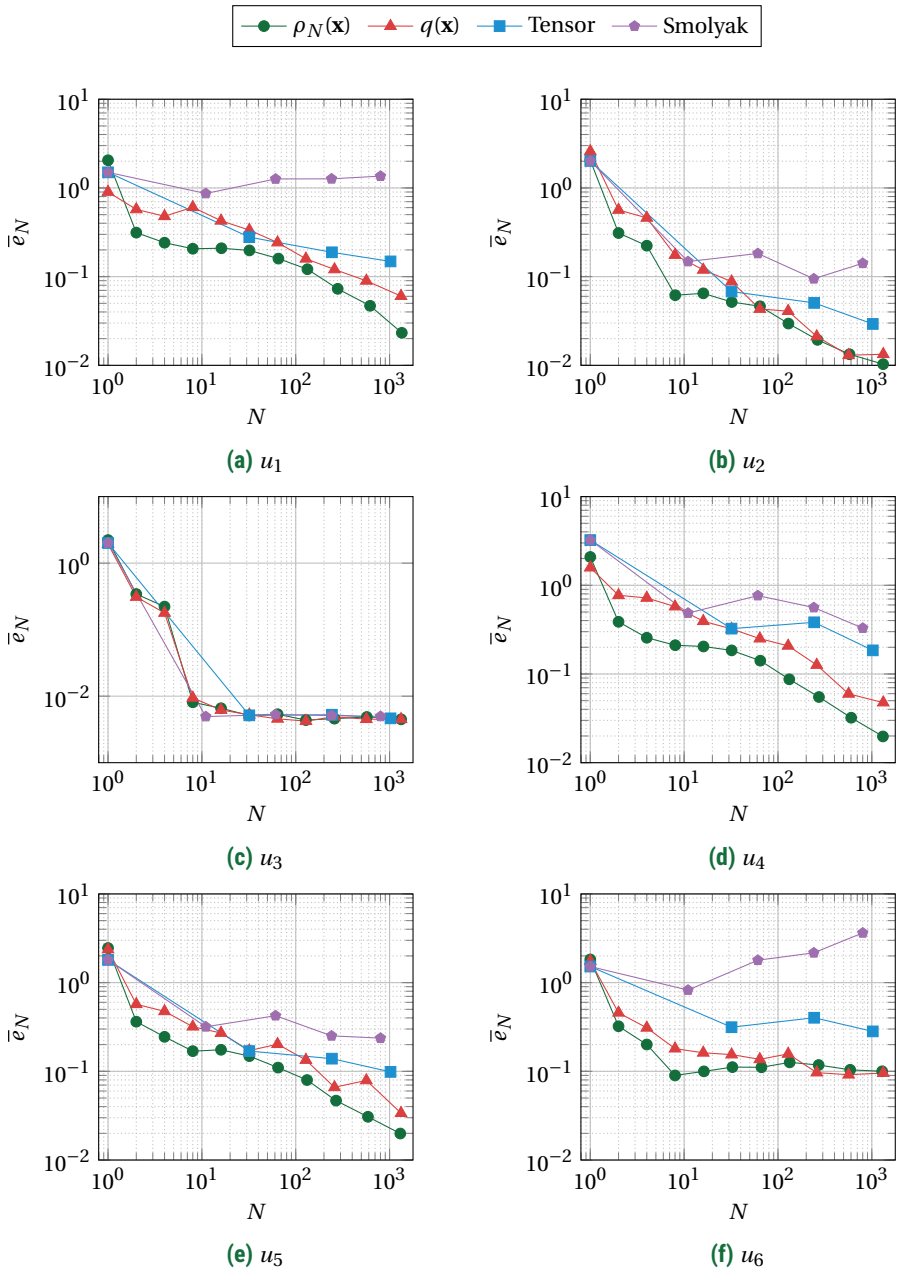


Figure 7.7: Convergence of the mean of the posterior determined using four different quadrature rules. The six functions under consideration are the Genz test functions.

The fifth Genz test function is only continuous (and not smooth), so should theoretically yield inferior performance. However, it is not differentiable only at a single point in the five-dimensional domain, so it is very unlikely that that point is of importance for the accuracy of the quadrature rules.

The sixth Genz test function is discontinuous and all rules do not yield a rapidly converging estimate. This is expected, since the quadrature rules are based on approximation using smooth functions. Notice that the convergence of the proposed adaptive procedure is significantly worse compared to the two-dimensional case considered previously, since detecting the discontinuity in the posterior using nearest neighbor interpolation in five dimensions requires significantly more model evaluations than in two dimensions. Nonetheless, the accuracy remains similar to the other quadrature rules.

Overall, the adaptive quadrature rule proposed here consistently performs better or on the same level as the other quadrature rules. This demonstrates that the proposed procedure works for non-informative posteriors, since it is rarely the case that the posterior of the Genz test functions in conjunction with the randomized parameters is informative. Intuitively one would expect that piecewise interpolation is in these cases not necessarily beneficial, but the results demonstrate that it does not deteriorate the performance.

The Smolyak sparse grid performs significantly worse than the other quadrature rules for u_1 and u_6. This happens due to the negative weights of the sparse grid, or more specifically, due to the large positive weights present in the Smolyak sparse grid. These large weights significantly deteriorate the accuracy for a small number of the 25 runs, which is reflected in the averaged errors presented in Figure 7.7. Notice that this effect can be significantly alleviated with an adaptive Smolyak sparse grid, as considered by Schillings and Schwab [159], though negative weights will remain present.

7.5.2. Transonic flow over an airfoil

The applicability of the approach to complex test cases is demonstrated by reconsidering the transonic flow over the RAE2822 airfoil. The problem under consideration is inferring Bayesian predictions of the pressure coefficient on the airfoil, incorporating calibrated closure parameters of the Spalart–Allmaras turbulence model using wind tunnel measurements. It is computationally expensive to determine an accurate numerical solution of this problem, so the usage of efficient calibration and prediction approaches is of importance.

This section is split into three parts. Firstly, the problem of modeling the transonic flow over an airfoil and the measurement data under consideration is briefly reconsidered (a more elaborate introduction can be found in Section 6.4.3). Secondly, the statistical model under consideration that relates the measurement data to the model output is discussed. The model incorporates hyperparameters, contrary to the case discussed in Chapter 6. Finally, Bayesian predictions of the flow around the airfoil are inferred.

Modeling the flow over an airfoil

The flow over the airfoil under consideration is modeled using the Reynolds-averaged Navier–Stokes (RANS) equations [192], in combination with the Spalart–Allmaras turbu-

lence model [170]. In this chapter the same variant as in Chapter 6 is considered, i.e. the model has the following parameters: σ, C_{b1}, C_{b2}, κ, C_{w2}, C_{w3}, C_{t3}, C_{t4}, C_{v1}, and C_{w1}. Here, C_{w1} is defined as follows:

$$C_{w1} = C_{b1}/\kappa^2 + (1 + C_{b2})/\sigma.$$

A major advantage of the approach discussed in this chapter is that the determined quadrature rule can directly be used to infer a Bayesian prediction of the flow incorporating the uncertainty of the closure coefficients. The costly model evaluations have already been performed during the construction of the rule. No Markov chain Monte Carlo routines are necessary to obtain the predictions.

Again SU2 [139] is employed to numerically solve the RANS equations and consequently obtain the pressure coefficient on the airfoil. The measurements from Cook et al. [36] are also being reused. We focus solely on Case 6 (see Cook et al. [36] for the other cases), with Reynolds number $6.5 \cdot 10^6$, Mach number 0.725, and angle of attack 2.92°. Recall that the measurements encompass $n = 103$ point measurements of the pressure coefficient on the surface of the airfoil.

Statistical model

The quantity of interest used for the calibration is the pressure coefficient on the surface of the airfoil. Following Edeling et al. [52], the parameters considered for calibration are $\boldsymbol{\vartheta} = (\kappa, \sigma, C_{b1}, C_{b2}, C_{v1}, C_{w2}, C_{w3})^{\mathrm{T}}$. The statistical model is the same model as considered in Chapter 6, with the sole difference that hyperparameters are incorporated in the calibration.

To this end, let $s \in [0,2]$ be the spatial parameter that runs from the trailing edge in anticlockwise direction over the airfoil. Then $u(\boldsymbol{\vartheta}; s)$ denotes the mapping that yields the pressure coefficient at s using the turbulence parameters $\boldsymbol{\vartheta}$. We denote the calibration parameters by $\boldsymbol{\vartheta}$ (instead of $\mathbf{x}$) to avoid confusion between $\boldsymbol{\vartheta}$ and the usual spatial coordinates x and y. A single evaluation of SU2 yields the pressure coefficient at all spatial coordinates in the discrete mesh for given $\boldsymbol{\vartheta}$ and we use linear interpolation to obtain the pressure coefficient at the measurement locations.

The statistical model is, following Kennedy and O'Hagan [95], as follows:

$$z_k = u(\boldsymbol{\vartheta}; s_k) + \delta(s_k) + \varepsilon_k, \text{ with } \varepsilon_k \sim \mathcal{N}(0, \widetilde{\sigma}^2) \text{ for } k = 1, \ldots, n. \tag{7.19}$$

Here, z_k denotes the measured pressure coefficient at location s_k. The random process $\delta(s) \sim \mathcal{N}(0, \mathrm{Cov}(s, s' \mid A, l))$ is a Gaussian process with zero mean and squared exponential covariance:

$$\mathrm{Cov}(s, s' \mid A, l) = A \exp\left[-\left(\frac{s - s'}{L 10^l}\right)^2\right].$$

The values of A and l are hyperparameters and are being inferred from calibration whereas L is a fixed parameter whose value is provided later. Measurement error is modeled by ε_k, which is Gaussian distributed with known standard deviation $\widetilde{\sigma} = 0.01$. This value is based on the measurement errors reported by Cook et al. [36].

The model from (7.19) yields the following likelihood:

$$q(\mathbf{z} \mid \boldsymbol{\vartheta}, A, l) \propto \exp\left[-\frac{1}{2}\mathbf{d}^{\mathrm{T}} C^{-1} \mathbf{d}\right], \tag{7.20}$$

with $\mathbf{d}$ the misfit and C the covariance matrix, i.e.

$$d_k = z_k - u(\boldsymbol{\vartheta}; s_k),$$
$$C = \tilde{\sigma} I + \Sigma, \text{ with } \Sigma_{i,j} = \mathrm{Cov}(s_i, s_j \mid A, l).$$

The prior of the turbulence closure parameters is the same as the one considered in the previous chapter, i.e. it is uniformly distributed with the following support:

κ	$[0.205, 0.615]$
σ	$[1/3, 1]$
C_{b1}	$[0.0678, 0.2033]$
C_{b2}	$[0.311, 0.933]$
C_{v1}	$[3.55, 10.65]$
C_{w2}	$[0.2, 0.4]$
C_{w3}	$[1, 3]$

The hyperparameters A and l are also being calibrated and therefore also require a prior. These are respectively $A \sim \mathcal{U}(0, 0.01)$ and $l \sim \mathcal{U}(0, 1)$. We choose $L = 0.01$, such that the maximal covariance length is significantly larger than the length of a single numerical cell on the surface of the airfoil. The computational model does not depend on A, l, and L, so given $\boldsymbol{\vartheta}$ the distribution of these parameters is fully known analytically.

By combining the prior with the likelihood from (7.20), Bayes' law yields the posterior:

$$q(\boldsymbol{\vartheta}, A, l \mid \mathbf{z}) \propto q(\mathbf{z} \mid \boldsymbol{\vartheta}, A, l)\, q(\boldsymbol{\vartheta})\, q(A)\, q(l).$$

We are not interested in the exact distribution of the hyperparameters and the evidence can be neglected by rescaling the quadrature rule weights. Hence we apply the adaptive implicit quadrature rule to the following distribution:

$$\rho(\boldsymbol{\vartheta}) = \iint q(\mathbf{z} \mid \boldsymbol{\vartheta}, A, l)\, q(\boldsymbol{\vartheta})\, q(A)\, q(l)\, \mathrm{d}A\, \mathrm{d}l.$$

Afterwards, the quadrature rule weights are scaled such that $\sum_{k=0}^{N} w_k = 1$ and the rule is used to infer Bayesian predictions, e.g. as derived in (7.2):

$$\mathbb{E}[\widehat{\mathbf{z}} \mid \mathbf{z}](s) = \int_{\Omega} u(\boldsymbol{\vartheta}, s)\, q(\boldsymbol{\vartheta} \mid \mathbf{z})\, \mathrm{d}\boldsymbol{\vartheta} \approx \sum_{k=0}^{N} u(\boldsymbol{\vartheta}_k, s)\, w_k, \text{ with } q(\boldsymbol{\vartheta} \mid \mathbf{z}) \propto \rho(\boldsymbol{\vartheta}).$$

Results

The quadrature rule constructed in this test case has polynomial degree 3, which constitutes a polynomial space with 120 basis polynomials. This number of polynomials is a good balance between accuracy in the predictions and overall running time. The quadrature rules are constructed with linearly increasing exactness, so at the i-th iteration a quadrature rule of degree i is constructed that incorporates all available model evaluations (up to iteration $i-1$).

The obtained rule incorporates an approximation of the posterior. There is no exact solution available, so it is not straightforward to assess the accuracy of the quadrature

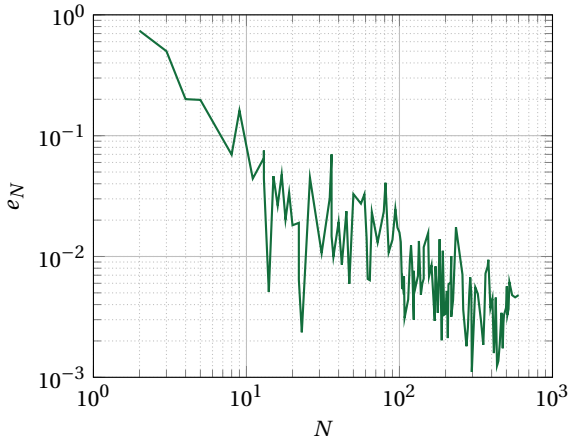


Figure 7.8: Convergence of the consecutive differences of the pressure coefficient on the RAE2822 airfoil.

rule. For this purpose, each iteration the mean of the pressure coefficient is determined and these are consecutively compared using the 2-norm, i.e. if N_i is the number of nodes of the quadrature rule after i iterations, then the error measure is

$$e_{N_i} = \|\mathcal{A}_{N_i} u - \mathcal{A}_{N_{i-1}} u\|_2.$$

We will denote this with a little abuse of notation simply as

$$e_N = \|\mathcal{A}_N u - \mathcal{A}_{N-1} u\|_2.$$

A similar error measure has been used in Section 4.4.2, where the flow over the NACA airfoil is considered. The obtained errors are depicted in Figure 7.8.

The quadrature rule clearly yields a converging mean and the error converges approximately linearly to zero. The oscillations are the result of the random sampling of the proposal distributions in combination with the fact that this test case was run only once. Obtaining a smoother decay would require repeating the experiment various times and averaging the obtained error (as done in the previous test cases), but this is too computationally costly in this case. In practical computations, the error decay can be assessed by regressing the convergence rate by means of least squares (i.e. by fitting analytic error decay).

Each iteration SU2 is evaluated for each additional quadrature rule node, so obtaining Bayesian predictions of the pressure coefficient is a straightforward post-processing step. The obtained predictions of the pressure coefficient are depicted in Figure 7.9. The measurement data, uncertainty bounds, and the deterministic pressure coefficient determined using the canonical turbulence coefficients are also depicted. It is clearly visible that the largest uncertainty occurs near the shock on top of the airfoil, which is in line with existing results on uncertain transonic flows around the RAE2822 airfoil, either using uncorrelated prescribed distributions [147, 195] or calibrated parameters in a Bayesian setting [18]. Notice that the number of evaluations of SU2 needed to infer these

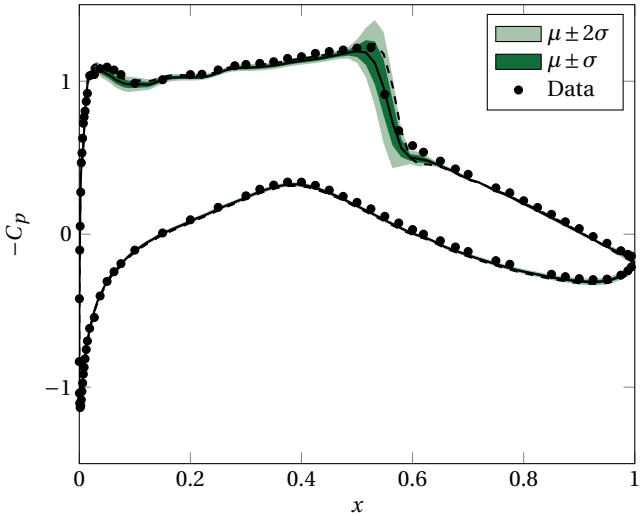


Figure 7.9: Mean (denoted by μ) and standard deviation (denoted by σ) of the pressure coefficients determined using Bayesian prediction and the method proposed in this chapter. The dashed line indicates the pressure coefficient obtained by using the canonical values of the turbulence closure coefficients.

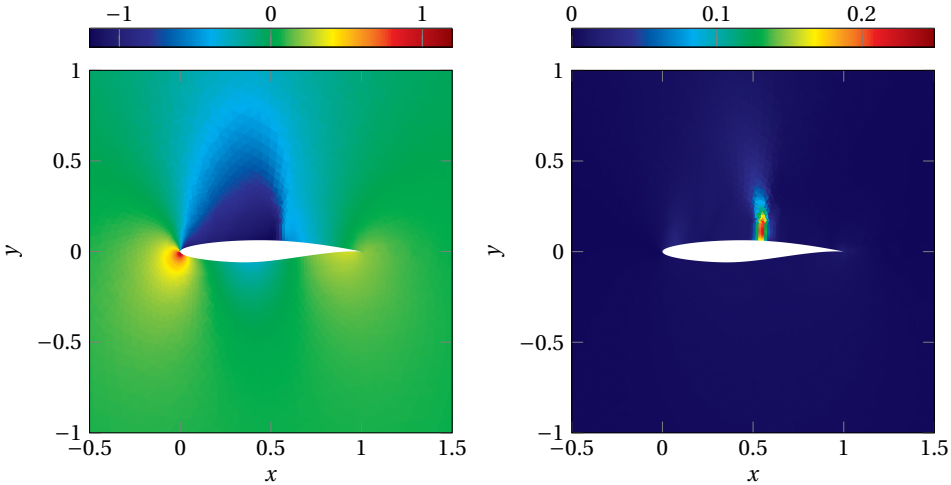
predictions is of similar order of magnitude as is common for uncertainty propagation with *prescribed* distributions. It is a significant reduction compared to commonly used Markov chain Monte Carlo methods [34, 51, 52].

The inferred prediction is similar to the prediction computed in Chapter 6 (see Figure 6.14 on page 155). In both cases the largest uncertainty is clearly originating from the shock on top of the airfoil, even though there are small differences. These differences are likely due to the calibration of hyperparameters and the usage of a significantly larger number of simulations.

`SU2` yields the full flow field around the airfoil and therefore it is also possible to predict the statistical moments of the pressure coefficient away from the airfoil surface, as depicted in Figure 7.10. We want to emphasize the importance of positive weights, since these ensure that the prediction of the variance is positive. Notice that the pressure coefficients depicted in these figures (i.e. away from the airfoil) have *not* been used in the calibration process, as only measurement data on the airfoil surface is available. So technically it not evident whether these results are correct or not. It is not straightforward to make general claims about this, since the specific properties (such as smoothness) of the model under consideration determine to a large extent whether it is viable to infer predictions of quantities using parameters that have been calibrated with a different quantity.

7.6. Conclusion

In this chapter a new methodology is proposed to construct quadrature rules with positive weights and high degree for the purpose of inferring Bayesian predictions. It



(a) Mean **(b)** Standard deviation

Figure 7.10: Predictions of pressure coefficients in the vicinity of the airfoil using calibrated coefficients.

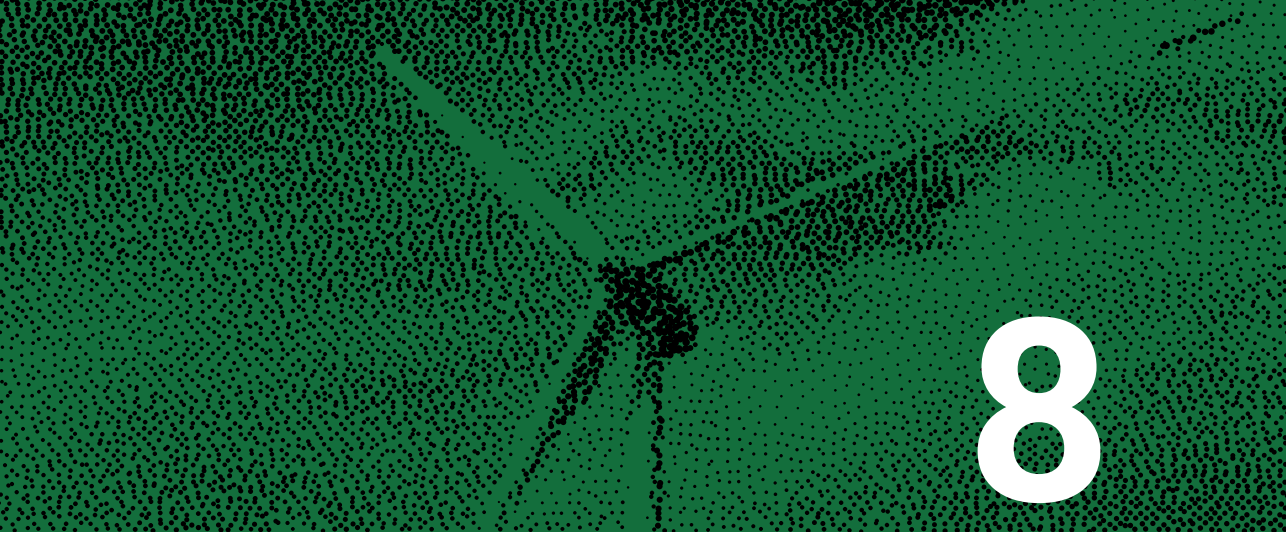
consists of two steps. Firstly a quadrature rule is constructed using an approximation of the posterior. For this step the implicit quadrature rule as introduced in Chapter 4 has been used. Secondly the approximation of the posterior is refined using all available model evaluations. For the second step nearest neighbor interpolation has been used. It has been demonstrated rigorously that our approach yields a quadrature rule whose error behaves like a quadrature rule with respect to the exact posterior for increasing number of nodes.

The performance of the quadrature rule has been demonstrated by calibrating the Genz test functions in a Bayesian framework. The results demonstrate that the quadrature rules consistently outperform (or perform on a par with) conventional numerical integration approaches. The performance gains are most significant if the likelihood is very informative, e.g. if its standard deviation is small. If, on the other hand, the posterior resembles approximately the prior, the performance is comparable to a quadrature rule determined using solely the prior.

The applicability of the approach to an expensive fluid dynamics model has been demonstrated by using the quadrature rule to calibrate the transonic flow over the RAE2822 airfoil. It has been demonstrated numerically that estimates of the rule converge and predictions of the mean and standard deviation of the pressure coefficient have been inferred. Significantly less model evaluations are necessary compared to existing Bayesian calibration approaches, while the results are in line with existing research.

There are various opportunities to further extend the approach. For example, the nearest neighbor interpolation procedure does not leverage smoothness in either the

distribution or the posterior. A major opportunity is replacing nearest neighbor interpolation by a more efficient interpolation approach, which could allow for more rapid convergence. This is however far from trivial, since our method requires that the obtained interpolant is a distribution.



8 Conclusions and future work

Many problems in wind energy, computational fluid dynamics, and other fields of engineering contain inherent randomness or uncertainty in the input parameters of the associated models. Assessing statistical quantities in these problems often involves calculating an integral with a non-trivial integrand. The computational cost of computing such integrals by random sampling is typically prohibitively large. Moreover, existing deterministic collocation approaches are usually inapplicable if the input parameters are correlated, dependent, or only known indirectly through the model. Therefore the key focus of this thesis has been to construct collocation approaches that are applicable regardless of the distribution of the input parameters, which makes these approaches particularly useful in problems related to wind energy.

The proposed collocation methods considered in this thesis, such as the implicit quadrature rule and its various extensions, are highly efficient methods to determine accurate statistics of computationally expensive models. The methods have been applied successfully to compute equivalent loads acting on an offshore wind turbine and it has been demonstrated both theoretically and numerically that converging estimates are obtained for sufficiently smooth functions.

The goal of this chapter is to briefly summarize this thesis, discuss the obtained results, draw conclusions, and provide options for future research.

8.1. Conclusions

The focus of this thesis is on the construction of efficient collocation methods. Based on the idea that Monte Carlo methods are too expensive to compute statistics of computationally expensive models and that other existing techniques require too stringent assumptions, novel methodologies and algorithms have been proposed to determine where to evaluate the computationally expensive model and how to post-process the obtained evaluations.

The proposed methods can be split into three categories, which follow roughly from the main objectives outlined in the introduction of this thesis. In Chapter 3 and 4 the focus was on uncertainty propagation, where the distribution of the parameters

of the model is fully characterized by a probability distribution or by a set of samples. In Chapter 5 the derived techniques were applied to a standardized load case, which specifies how to calculate equivalent loads acting on a wind turbine. Finally, in Chapter 6 and 7 the focus shifted to the case where the distribution of the parameters cannot be fully characterized beforehand, since it directly depends on measurement data and the model. Such distributions are often obtained in Bayesian calibration and prediction problems.

8.1.1. Uncertainty propagation

Uncertainty propagation encompasses the problem of assessing the statistics of a computationally expensive model with random parameters. For this purpose, a family of quadrature rules has been proposed in this thesis. The quadrature rules follow from a mathematical framework that describes various algorithms that can be used to modify quadrature rules. All modifications are such that the quadrature rules remain accurate, or more specifically, such that the quadrature rules keep positive weights and remain interpolatory. By exploiting this framework, the implicit quadrature rule was derived, which is a quadrature rule with positive weights obtained as a subset of a large number of samples.

The proposed numerical integration techniques are very suitable to infer moments (or other integral quantities) of computationally expensive models with random parameters. Any set of samples defined on any domain can be used to construct the quadrature rules. Moreover the computational cost of the algorithms does not scale with the dimension of the random space. High convergence rates were obtained for smooth functions and no deterioration of convergence was observed for non-smooth functions. These properties have been verified both theoretically and numerically.

A limitation of the quadrature rule framework discussed in this thesis is that it must be initialized with an existing quadrature rule with positive weights. This rule does not need to be based on polynomials. For example, equally weighted random samples can be used, which forms the key idea of the implicit quadrature rule. Using random samples limits the accuracy of the quadrature rules to the accuracy of the set of samples but allows to compute quadrature rules with respect to distributions that are only known by samples or measurements.

The integration error of all constructed quadrature rules is heavily influenced by the usage of such an initial quadrature rule. To illustrate the impact of all sources of error on the integration error, the absolute integration error has been split in several components: the quadrature error, the sampling error (Chapter 4 and 7), the seed error (Chapter 5), and the interpolation error (Chapter 7). This provides insight into the decay of the integration error, but no techniques have been discussed to balance these errors and thereby optimizing the computational cost by distributing computational power equally across all sources of error.

The quadrature rule framework and the implicit quadrature rule are based on a solid mathematical foundation that describe the accuracy of the constructed rules. The numerical stability of constructing the rules has not been considered extensively in this thesis, but is a non-trivial and challenging element, especially if quadrature rules of large numbers of nodes are considered. For the purposes of this thesis, the algorithms

proposed in this thesis are sufficient, but for larger scale applications more involved methods might be necessary.

It remains notoriously difficult to directly construct a sequence of nested quadrature rules with positive weights given an arbitrary distribution. Although the approaches proposed in this thesis circumvent the direct construction of rules by considering existing rules, they could be significantly improved if such techniques would be available. Based among others on the fact that any arbitrary sized interpolatory quadrature rule can be used to construct a finitely sized sequence of nested quadrature rules, there are strong theoretical indications that sequences of nested quadrature rules with positive weights can be constructed given an arbitrary basis, distribution, and domain of definition. However, an algorithm to construct such rules has not been derived.

8.1.2. Wind turbine load calculation

One of the main objectives of this thesis is to assess the loads acting on a wind turbine more efficiently. This is non-trivial with existing methods, since the distributions involved in load calculations are often strongly correlated or only known by measurements. The implicit quadrature rule is tuned specifically to these restrictions and can be used straightforwardly to assess standardized fatigue load cases.

To demonstrate the applicability of the quadrature rules to load cases, the damage equivalent loads acting on the NREL 5MW turbine have been assessed. Equivalent loads are defined as an integral weighted with a correlated probability density function that should ideally be inferred from measurements. Numerical integration techniques that can handle such distributions, such as the implicit quadrature rule, form promising alternatives to existing methods such as binning.

The obtained damage equivalent loads are of the same order of magnitude as obtained by the standardized technique, although a significantly smaller number of model runs is necessary. The key properties of quadrature rules, such as high convergence rates for smooth functions, again manifested themselves in this test case. The accuracy of the methods has been validated by comparing consecutive approximations with different fidelity of the equivalent loads.

Even though the implicit quadrature rule directly competes with binning, this does not necessarily imply that the obtained estimations are accurate. A large validation test case, for example a comparison of the obtained estimations with a high fidelity Monte Carlo estimation, could not be conducted using the hardware employed for the simulations done in this thesis. However, such a validation test case is necessary before applying these methods to “real” wind turbines and offshore wind farms.

As noticed above, the implicit quadrature rule is only applicable if a set of samples (or measurements) is available. For the computations of the design load case considered in this thesis a large number of measurements conducted at the North Sea is employed. Therefore the proposed procedure is very suitable for assessing the life time of an existing wind turbine design at an offshore site or for designing a wind turbine specifically with an offshore site in mind (this is standardized as the S-class [85]). However, offshore measurements are not always available during the design phase of a wind turbine and are costly to obtain.

8.1.3. Bayesian calibration and prediction

In many computational methods it is the case that exact values of parameters are unknown or non-existent and no closed-form distribution is provided describing the variability in these parameters. For example, the Reynolds-averaged Navier Stokes equations require a closure model to form a closed system of differential equations and closure models are often accompanied by closure or fitting parameters that require calibration. Bayesian model calibration is a systematic way to model the uncertainty of these parameters by describing the distribution as a posterior of a calibration procedure in a Bayesian framework. By propagating the posterior through the model, predictions of the model incorporating uncertainty can be inferred.

The key challenge, compared to "vanilla" uncertainty propagation, is that the posterior explicitly depends on the model. To alleviate the high computational cost associated with inferring statistics of such a posterior, two approaches have been proposed in this thesis. The first approach (discussed in Chapter 6) consists of constructing a surrogate of the model which is tailored specifically to the available knowledge of the posterior. This procedure shows its true strength for models which are smooth and analytic, yielding high convergence rates in that case. The second approach (discussed in Chapter 7) reuses the quadrature rules derived for propagation of uncertainty and consists of iteratively constructing a quadrature rule that incorporates the posterior. Since it is a numerical integration technique, it is especially suitable for inferring Bayesian predictions. It works for any model that can be integrated well using a quadrature rule, although it does not immediately reveal the structure of the posterior itself.

The two proposed procedures have been applied to the calibration of the turbulence closure parameters, where the goal was to calibrate a computational fluid dynamics model using wind tunnel measurements. In both cases accurate predictions could be inferred with a small number of model evaluations (compared to conventional Monte Carlo approaches). It demonstrates that these approaches are viable alternatives to existing methods for Bayesian inference with computationally expensive models.

Without some basic knowledge of the model, such as its smoothness properties, it is not straightforward to predict which approach is more suitable for the problem at hand. If no assumptions can be made about the model under consideration, the approach based on integration is favorable due to its generality, but this approach converges significantly slower compared to the adaptive interpolation approach if the model is smooth and analytic. Moreover a disadvantage of this approach is that only integral quantities can be accurately computed.

The core of Bayesian model calibration and prediction consists of a statistical model that describes the relation between the output of the model and the measurement data. If the statistical model is chosen incorrectly, this has immediate effect on the results obtained. The computed predictions might be overconfident or be uninformative. Choosing a correct statistical model is therefore essential for any Bayesian analysis, but this step has not been discussed extensively in this thesis.

8.2. Future work

A key goal of this thesis is to propose efficient methods to assess the loads acting on an offshore wind turbine. Various approaches have been proposed and their effectiveness

has been demonstrated. Based on these approaches, there are numerous opportunities to apply these methodologies to different scenarios, extent the algorithms, or use alternative approaches. The recommendations and suggestions for future research directions are split into two categories. In Section 8.2.1 the methods to assess uncertainty (both propagation and calibration) are considered, mostly from a mathematical point of view. In Section 8.2.2 the wind turbine load case calculations considered in this thesis are discussed.

8.2.1. Uncertainty quantification

As concluded above, all multivariate numerical integration methodologies proposed in this thesis are constructed using sample sets. Even though the number of samples can be chosen independently of the number of nodes of the quadrature rule, the accuracy of the quadrature rule cannot supersede the accuracy of the set of samples. It follows from the quadrature rule framework derived in this thesis that it is non-trivial to directly accommodate the exact moments of the distributions, but if possible, the accuracy of the quadrature rule can be improved by incorporating these values. Based on the results of this thesis, it is recommended to search for an algorithm that can construct quadrature rules with positive weights that integrate an arbitrary number of basis functions. Consequently a nested sequence can be obtained by removing nodes from this rule, as outlined in Chapter 3.

The quadrature rules have solely been used for *non-intrusive* uncertainty quantification. It is only assumed that general characteristics of the model are known and no modifications of existing implementations are necessary. This is an advantage, since this allows to apply the proposed methods to many different problems with varying characteristics. The performance of the approaches can possibly be improved by incorporating knowledge about the model in the collocation approaches.

In this thesis Leja nodes have been used to construct an interpolating polynomial. It has been shown numerically that the Lebesgue constant of these nodes, which is an indicator of their accuracy, grows sublinearly. Proving this result theoretically allows for broader usage of these nodes and it is a recommended future research direction.

Throughout this thesis, the Genz test functions have been used to demonstrate the efficiency of the proposed quadrature methods. These functions are commonly used for this purpose. On the other hand, there does not exist a canonical test case for Bayesian calibration and prediction. In this thesis the calibration of the turbulence closure coefficients has been considered as canonical test case, but it would be beneficial and recommended for the field of uncertainty quantification in general if test cases for the purpose of Bayesian calibration become standardized and freely available, similarly to the test functions used to test numerical integration techniques.

Many techniques in uncertainty quantification, including the ones used in this thesis, originate from other fields of applied mathematics, such as approximation theory. Essentially the goal is to approximate a function or a linear operator (such as an integral) acting on the function. Hoever, the methods used in uncertainty quantification use different nomenclature, even though they are based on the same techniques. For instance, a polynomial chaos expansion is equivalent to a pseudo-spectral expansion and stochastic collocation is equivalent to numerical integration or simply “conven-

tional" collocation. This inconvenient usage of nomenclature might yield confusion in the future and could hinder sharing improvements found in both fields. It is not recommended to change all nomenclature retroactively, but it is certainly beneficial for the field of uncertainty quantification if new nomenclature would be introduced carefully.

8.2.2. Load case calculations

In this thesis, a single load case has been used to demonstrate the applicability of quadrature rules to load case calculations. The promising results demonstrate the prospect of the quadrature rule. It should be straightforward to consider other fatigue load cases, but it is significantly more challenging to consider ultimate load cases. Ultimate load cases require a non-trivial extrapolation step to account for rare events that can severely affect the life time of the wind farm. Extending the proposed collocation framework to ultimate loads is an important research direction to further study the usage of numerical integration techniques in load case calculations.

The proposed approach to assess load cases has been validated by comparing it to the conventional approach ("binning"). A major disadvantage of this comparison is that binning also contains an error, so it is difficult to determine the source of possible discrepancy between the obtained results. Ideally, the accuracy of the quadrature rule would be assessed by means of a reference value, but obtaining such a value is computationally very demanding. Moreover, conducting measurements is usually very expensive. It would be beneficial for the development of new techniques if the IEC standard would describe a reference test case (including a description of a wind turbine) and reference values of the quantities described by the load cases. These values can consequently be used to quantitatively compare the different methods (such as quadrature rules and binning, but possibly also others).

Current approaches used to assess loads that act on a wind turbine are relatively straightforward, compared with the recent approaches that are available in the field of approximation theory. This is not surprising, since standardized methods should be thoroughly tested. In recent versions of the IEC standard, methods from the field of uncertainty quantification are briefly mentioned and discussed. Further enhancing these methods and standardizing their usage would allow for faster load case assessment, and therefore ultimately to cheaper development of offshore wind farms.

The Bayesian calibration and prediction approaches discussed in this thesis can be used straightforwardly to calibrate a wind turbine model and infer predictions of the equivalent loads that incorporate besides a varying environment also the error in the model to a certain extent. However, calibration of the model requires measurements of the forces acting on a turbine, which are currently not available for research purposes. This is a limitation for all data-driven approaches in wind energy problems. Making such measurements available facilitates the verification and development of computational methods tailored to wind energy.

References

[1] G. N. Absi and S. Mahadevan. Calibration of system parameters under model uncertainty. In H. S. Atamturktur, B. Moaveni, C. Papadimitriou, and T. Schoenherr, editors, *Model Validation and Uncertainty Quantification*. Volume 3, pages 1–10. Springer International Publishing, 2014. DOI: 10.1007/978-3-319-04552-8_1 (cited on page 124).

[2] P. Agarwal and L. Manuel. Extreme loads for an offshore wind turbine using statistical extrapolation from limited field data. *Wind Energy*, **11**(6):673–684, 2008. DOI: 10.1002/we.301 (cited on page 100).

[3] S. Ahn, A. Korattikara, and M. Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *Proceedings of the 29th International Conference on Machine Learning*, 2012 (cited on page 158).

[4] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, **45**(3):1005–1034, 2007. DOI: 10.1137/050645142 (cited on page 100).

[5] C. Bayer and J. Teichmann. The proof of Tchakaloff's theorem. *Proceedings of the American Mathematical Society*, **134**(10):3035–3041, 2006. DOI: 10.1090/s0002-9939-06-08249-9 (cited on page 69).

[6] M. A. Beaumont, W. Zhang, and D. J. Balding. Approximate Bayesian computation in population genetics. *Genetics*, **162**(4):2025–2035, 2002 (cited on page 124).

[7] J.-P. Berrut and L. N. Trefethen. Barycentric Lagrange interpolation. *SIAM Review*, **46**(3):501–517, 2004. DOI: 10.1137/s0036144502417715 (cited on pages 39, 54, 127).

[8] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Elsevier, 1982. ISBN: 978-0-12-093480-5. DOI: 10.1016/c2013-0-10366-2 (cited on page 112).

[9] G. Biau and L. Devroye. *Lectures on the Nearest Neighbor Method*. Springer, 2015. ISBN: 978-3-319-25386-2. DOI: 10.1007/978-3-319-25388-6 (cited on page 167).

[10] I. Bilionis and N. Zabaras. Solution of inverse problems with limited forward solver evaluations: a Bayesian perspective. *Inverse Problems*, **30**(1):015004, 2013. DOI: 10.1088/0266-5611/30/1/015004 (cited on page 158).

[11] A. Birolleau, G. Poëtte, and D. Lucor. Adaptive Bayesian inference for discontinuous inverse problems, application to hyperbolic conservation laws. *Communications in Computational Physics*, **16**(1):1–34, 2014. DOI: 10.4208/cicp.240113.071113a (cited on pages 124, 125, 140, 158).

[12] *BLADED Theory Manual*. Version 4.6. DNV GL - Energy. 2014 (cited on page 105).

[13] L. M. M. van den Bos. The implicit quadrature rule. Zenodo, Software, 2019. DOI: 10.5281/zenodo.3234434 (cited on page 67).

[14] L. M. M. van den Bos, W. A. A. M. Bierbooms, A. Alexandre, B. Sanderse, and G. J. W. van Bussel. Fatigue design load calculations of the offshore NREL 5MW benchmark turbine using quadrature rule techniques. *To appear in Wind Energy*, 2019. arXiv: 1904.07021 [cs.CE] (cited on page 99).

[15] L. M. M. van den Bos, B. Koren, and R. P. Dwight. Non-intrusive uncertainty quantification using reduced cubature rules. *Journal of Computational Physics*, **332**:418–445, 2017. DOI: 10.1016/j.jcp.2016.12.011. arXiv: 1905.06177 [math.NA] (cited on pages 27, 28, 32, 37, 69, 76, 100).

[16] L. M. M. van den Bos and B. Sanderse. A geometric approach for the addition of nodes to an interpolatory quadrature rule with positive weights. *Under review*, 2019. arXiv: 1902.07477 [math.NA] (cited on page 27).

[17] L. M. M. van den Bos, B. Sanderse, and W. A. A. M. Bierbooms. Adaptive sampling-based quadrature rules for efficient Bayesian prediction. *Under review*, 2019. arXiv: 1907.08418 [math.NA] (cited on page 157).

[18] L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes. *Communications in Computational Physics*, **27**(1):33–69, 2020. DOI: 10.4208/cicp.oa-2018-0218. arXiv: 1802.02035 [math.NA] (cited on pages 68, 95, 123, 186).

[19] L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Generating nested quadrature rules with positive weights based on arbitrary sample sets. *To appear in SIAM/ASA Journal on Uncertainty Quantification*, 2018. arXiv: 1809.09842 [math.NA] (cited on page 67).

[20] L. M. M. van den Bos, B. Sanderse, L. Blonk, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Efficient ultimate load estimation for offshore wind turbines using interpolating surrogate models. *Journal of Physics: Conference Series*, **1037**:062017, 2018. DOI: 10.1088/1742-6596/1037/6/062017 (cited on page 100).

[21] C. Botts, W. Hörmann, and J. Leydold. Transformed density rejection with inflection points. *Statistics and Computing*, **23**(2):251–260, 2011. DOI: 10.1007/s11222-011-9306-4 (cited on page 158).

[22] R. Bourquin. Exhaustive search for higher-order Kronrod–Patterson Extensions. Technical report 2015-11, ETH Zürich, 2015 (cited on page 28).

[23] L. Brandolini, C. Choirat, L. Colzani, G. Gigante, R. Seri, and G. Travaglini. Quadrature rules and distribution of points on manifolds. *Annali della Scuola Normale Superiore - Classe di Scienze*, **13**(4):889–923, 2014. DOI: 10.2422/2036-2145.201103_007 (cited on page 17).

[24] H. Brass and K. Petras. *Quadrature Theory: The Theory of Numerical Integration on a Compact Interval*. R. L. Cohen, J. S. Ellenberg, M. A. Singer, and B. Sudakov, editors, volume 178 of *Mathematical Surveys and Monographs*. American Mathematical Society, 2011. ISBN: 978-0-8218-5361-0. DOI: 10.1090/surv/178 (cited on pages 16, 17, 19, 28, 30, 31, 72, 100, 158, 159).

[25] P. Bratley and B. L. Fox. ALGORITHM 659: implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software*, **14**(1):88–100, 1988. DOI: 10.1145/42288.214372 (cited on page 11).

[26] L. Brutman. Lebesgue functions for polynomial interpolation – a survey. *Annals of Numerical Mathematics*, **4**:111–127, 1997 (cited on page 13).

[27] M. D. Buhmann. Radial basis functions. *Acta Numerica*, **9**:1–38, 2000. DOI: 10.1017/s0962492900000015 (cited on page 168).

[28] T. Burton, D. Sharpe, N. Jenkins, and E. Bossanyi. *Wind Energy Handbook*. Wiley, 2001. ISBN: 978-0-471-48997-9. DOI: 10.1002/0470846062 (cited on page 105).

[29] T. Butler, L. Graham, S. Mattis, and S. Walsh. A measure-theoretic interpretation of sample based numerical integration with applications to inverse and prediction problems under uncertainty. *SIAM Journal on Scientific Computing*, **39**(5):A2072–A2098, 2017. DOI: 10.1137/16m1063289 (cited on pages 160, 168).

[30] T. Butler, J. Jakeman, and T. Wildey. Convergence of probability densities using approximate models for forward and inverse problems in uncertainty quantification. *SIAM Journal on Scientific Computing*, **40**(5):A3523–A3548, 2018. DOI: 10.1137/18m1181675 (cited on page 140).

[31] R. E. Caflisch. Monte Carlo and quasi-Monte Carlo methods. *Acta Numerica*, **7**:1–49, 1998. DOI: 10.1017/s0962492900002804 (cited on pages 10, 11, 77, 100, 111, 158, 176).

[32] J.-P. Calvi and M. Phung Van. On the Lebesgue constant of Leja sequences for the unit disk and its applications to multivariate interpolation. *Journal of Approximation Theory*, **163**(5):608–622, 2011. DOI: 10.1016/j.jat.2011.02.001 (cited on page 138).

[33] D. Chauveau and P. Vandekerkhove. Improving convergence of the Hastings–Metropolis algorithm with an adaptive proposal. *Scandinavian Journal of Statistics*, **29**(1):13–29, 2002. DOI: 10.1111/1467-9469.00064 (cited on page 124).

[34] S. H. Cheung, T. A. Oliver, E. E. Prudencio, S. Prudhomme, and R. D. Moser. Bayesian uncertainty analysis with applications to turbulence modeling. *Reliability Engineering & System Safety*, **96**(9):1137–1149, 2011. DOI: 10.1016/j.ress.2010.09.013 (cited on pages 124, 151, 154, 187).

[35] C. W. Clenshaw and A. R. Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, **2**(1):197–205, 1960. DOI: 10.1007/bf01386223 (cited on pages 19, 28, 61, 109).

[36] P. H. Cook, M. A. McDonald, and M. C. P. Firmin. Aerofoil RAE 2822 – pressure distributions, and boundary layer and wake measurements. In *Experimental Data Base for Computer Program Assessment*, AGARD Advisory Report No. 138, A6-1–A6-77. North Atlantic Treaty Organization, 1979. ISBN: 978-92-835-1323-0 (cited on pages 149, 150, 152, 184).

[37] R. Cools. Constructing cubature formulae: the science behind the art. *Acta Numerica*, **6**:1–54, 1997. DOI: 10.1017/s0962492900002701 (cited on page 18).

[38] R. Cools and A. Haegemans. On the construction of multi-dimensional embedded cubature formulae. *Numerische Mathematik*, **55**(6):735–745, 1989. DOI: 10.1007/bf01389339 (cited on page 32).

[39] S. L. Cotter, M. Dashti, and A. M. Stuart. Approximation of Bayesian inverse problems for PDEs. *SIAM Journal on Numerical Analysis*, **48**(1):322–345, 2010. DOI: 10.1137/090770734 (cited on page 68).

[40] K. Csilléry, M. G. B. Blum, O. E. Gaggiotti, and O. François. Approximate Bayesian computation (ABC) in practice. *Trends in Ecology & Evolution*, **25**(7):410–418, 2010. DOI: 10.1016/j.tree.2010.04.001 (cited on page 124).

[41] G. Dahlquist and Å. Björck. *Numerical Methods in Scientific Computing, Volume I*. Society for Industrial and Applied Mathematics, 2008. ISBN: 978-0-89871-644-3. DOI: 10.1137/1.9780898717785 (cited on page 28).

[42] P. J. Davis. A construction of nonnegative approximate quadratures. *Mathematics of Computation*, **21**(100):578–582, 1967. DOI: 10.1090/s0025-5718-1967-0222534-4 (cited on pages 69, 73).

[43] A. P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *The Annals of Mathematical Statistics*, **38**(2):325–339, 1967. DOI: 10.1214/aoms/1177698950 (cited on page 22).

[44] G. Derflinger, W. Hörmann, and J. Leydold. Random variate generation by numerical inversion when only the density is known. *ACM Transactions on Modeling and Computer Simulation*, **20**(4):1–25, 2010. DOI: 10.1145/1842722.1842723 (cited on page 158).

[45] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag New York Inc., 1986. ISBN: 978-1-4613-8645-2. DOI: 10.1007/978-1-4613-8643-8 (cited on page 70).

[46] A. Doostan and H. Owhadi. A non-adapted sparse approximation of PDEs with stochastic inputs. *Journal of Computational Physics*, **230**(8):3015–3034, 2011. DOI: 10.1016/j.jcp.2011.01.002 (cited on pages 100, 101).

[47] T. A. Driscoll, N. Hale, and L. N. Trefethen. *Chebfun Guide*. Pafnuty Publications, 2014 (cited on page 62).

[48] Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Review*, **41**(4):637–676, 1999. DOI: 10.1137/s0036144599352836 (cited on pages 160, 168).

[49] M. S. Ebeida, S. A. Mitchell, L. P. Swiler, V. J. Romero, and A. A. Rushdi. POF-darts: geometric adaptive sampling for probability of failure. *Reliability Engineering & System Safety*, **155**:64–77, 2016. DOI: 10.1016/j.ress.2016.05.001 (cited on page 168).

[50] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. SU2: an open-source suite for multiphysics simulation and design. *AIAA Journal*, **54**(3):828–846, 2016. DOI: 10.2514/1.j053813 (cited on page 94).

[51] W. N. Edeling, P. Cinnella, and R. P. Dwight. Predictive RANS simulations via Bayesian model-scenario averaging. *Journal of Computational Physics*, **275**:65–91, 2014. DOI: 10.1016/j.jcp.2014.06.052 (cited on pages 151, 154, 187).

[52] W. N. Edeling, P. Cinnella, R. P. Dwight, and H. Bijl. Bayesian estimates of parameter variability in the $k-\epsilon$ turbulence model. *Journal of Computational Physics*, **258**:73–94, 2014. DOI: 10.1016/j.jcp.2013.10.027 (cited on pages 24, 124, 160, 184, 187).

[53] W. N. Edeling, R. P. Dwight, and P. Cinnella. Simplex-stochastic collocation method with improved scalability. *Journal of Computational Physics*, **310**:301–328, 2016. DOI: 10.1016/j.jcp.2015.12.034 (cited on page 168).

[54] A. Eggels, D. Crommelin, and J. Witteveen. Clustering-based collocation for uncertainty propagation with multivariate dependent inputs. *International Journal for Uncertainty Quantification*, **8**(1):43–59, 2018. DOI: 10.1615/int.j.uncertaintyquantification.2018020215 (cited on pages 10, 69).

[55] M. S. Eldred, L. P. Swiler, and G. Tang. Mixed aleatory-epistemic uncertainty quantification with stochastic expansions and optimization-based interval estimation. *Reliability Engineering & System Safety*, **96**(9):1092–1113, 2011. DOI: 10.1016/j.ress.2010.11.010 (cited on page 22).

[56] M. S. Eldred and L. P. Swiler. Efficient Algorithms for Mixed Aleatory-Epistemic Uncertainty Quantification with Application to Radiation-Hardened Electronics. Technical report SAND2009–5805, Sandia National Laboratories, 2009 (cited on page 22).

[57] C. M. Engelen. *The nonlinear effect of combining uncertainties on the energy yield of an offshore wind farm*. Master's thesis, Delft University of Technology, 2015 (cited on page 10).

[58] J. Feinberg, V. G. Eck, and H. P. Langtangen. Multivariate polynomial chaos expansions with dependent variables. *SIAM Journal on Scientific Computing*, **40**(1):A199–A223, 2018. DOI: 10.1137/15m1020447 (cited on page 68).

[59] D. Foti, X. Yang, and F. Sotiropoulos. Uncertainty quantification of infinite aligned wind farm performance using non-intrusive polynomial chaos and a distributed roughness model. *Wind Energy*, **20**(6):945–958, 2016. DOI: 10.1002/we.2072 (cited on pages 15, 100).

[60] I. M. Franck and P. S. Koutsourelakis. Sparse variational Bayesian approximations for nonlinear inverse problems: applications in nonlinear elastography. *Computer Methods in Applied Mechanics and Engineering*, **299**:215–244, 2016. DOI: 10.1016/j.cma.2015.10.015 (cited on page 68).

[61] R. Franke. Scattered data interpolation: tests of some methods. *Mathematics of Computation*, **38**(157):181–200, 1982. DOI: 10.1090/s0025-5718-1982-0637296-4 (cited on page 167).

[62] K. Freudenreich and K. Argyriadis. Wind turbine load level based on extrapolation and simplified methods. *Wind Energy*, **11**(6):589–600, 2008. DOI: 10.1002/we.279 (cited on page 100).

[63] F. N. Fritsch and R. E. Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, **17**(2):238–246, 1980. DOI: 10.1137/0717021 (cited on page 168).

[64] W. Gautschi. On inverses of Vandermonde and confluent Vandermonde matrices. *Numerische Mathematik*, **4**(1):117–123, 1962. DOI: 10.1007/bf01386302 (cited on page 50).

[65] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, third edition, 2013. ISBN: 978-1-4398-4095-5 (cited on pages 22, 126, 158).

[66] A. Genz. Testing multidimensional integration routines. In *Proceedings of the International Conference on Tools, Methods and Languages for Scientific and Engineering Computation*, pages 81–94. Elsevier North–Holland, 1984 (cited on pages 61, 62, 87, 88, 89, 113, 176, 178, 180).

[67] A. Genz and B. D. Keister. Fully symmetric interpolatory rules for multiple integrals over infinite regions with Gaussian weight. *Journal of Computational and Applied Mathematics*, **71**(2):299–309, 1996. DOI: 10.1016/0377-0427(95)00232-4 (cited on page 28).

[68] T. Gerstner and M. Griebel. Numerical integration using sparse grids. *Numerical Algorithms*, **18**(3):209–232, 1998. DOI: 10.1023/a:1019129717644 (cited on pages 100, 101).

[69] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, **79**(11):1309–1331, 2009. DOI: 10.1002/nme.2579 (cited on page 94).

[70] R. G. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer New York, 1991. ISBN: 978-1-4612-7795-8. DOI: 10.1007/978-1-4612-3094-6 (cited on pages 7, 8, 12).

[71] M. B. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, **56**(3):607–617, 2008. DOI: 10.1287/opre.1070.0496 (cited on page 11).

[72] Global wind statistic 2015, Global Wind Energy Council, 2016 (cited on page 1).

[73] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996. ISBN: 978-0-8018-5413-2 (cited on page 128).

[74] G. H. Golub and J. H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation*, **23**(106):221–230, 1969. DOI: 10.1090/s0025-5718-69-99647-1 (cited on pages 18, 27, 30, 59, 61, 109, 159).

[75] P. Graf, K. Dykes, R. Damiani, J. Jonkman, and P. Veers. Adaptive stratified importance sampling: hybridization of extrapolation and importance sampling Monte Carlo methods for estimation of wind turbine extreme loads. *Wind Energy Science*, **3**(2):475–487, 2018. DOI: 10.5194/wes-3-475-2018 (cited on page 100).

[76] L. A. Grzelak, J. A. S. Witteveen, M. Suárez-Taboada, and C. W. Oosterlee. The stochastic collocation Monte Carlo sampler: highly efficient sampling from 'expensive' distributions. *Quantitative Finance*, **19**(2):339–356, 2018. DOI: 10.1080/14697688.2018.1459807 (cited on page 168).

[77] A. Guessab and G. Schmeisser. Construction of positive definite cubature formulae and approximation of functions via Voronoi tessellations. *Advances in Computational Mathematics*, **32**(1):25–41, 2008. DOI: 10.1007/s10444-008-9080-9 (cited on pages 160, 168).

[78] R. Günttner. Evaluation of Lebesgue constants. *SIAM Journal on Numerical Analysis*, **17**(4):512–520, 1980. DOI: 10.1137/0717043 (cited on page 13).

[79] T. Hasegawa, H. Sugiura, and T. Torii. Positivity of the weights of extended Clenshaw–Curtis quadrature rules. *Mathematics of Computation*, **60**(202):719–734, 1993. DOI: 10.1090/s0025-5718-1993-1176710-2 (cited on page 28).

[80] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**(1):97–109, 1970. DOI: 10.1093/biomet/57.1.97 (cited on pages 24, 70, 124, 126, 158).

[81] J. Hewitt and J. A. Hoeting. Approximate Bayesian inference via sparse grid quadrature evaluation for hierarchical models, 2019. arXiv: 1904.07270 [stat.CO] (cited on page 159).

[82] N. J. Higham. The numerical stability of barycentric Lagrange interpolation. *IMA Journal of Numerical Analysis*, **24**(4):547–556, 2004. DOI: 10.1093/imanum/24.4.547 (cited on page 127).

[83] G. Iaccarino, G. Petrone, J. Witteveen, D. Quagliarella, C. D. Nicola, and J. Axerio-Cilies. Wind turbine optimization under uncertainty with high performance computing. In *29th AIAA Applied Aerodynamics Conference*. American Institute of Aeronautics and Astronautics, 2011. DOI: 10.2514/6.2011-3806 (cited on page 15).

[84] B. A. Ibrahimoglu. Lebesgue functions and Lebesgue constants in polynomial interpolation. *Journal of Inequalities and Applications*, **2016**(1):93, 2016. DOI: 10.1186/s13660-016-1030-3 (cited on pages 13, 28, 125, 136, 138).

[85] IEC. Wind turbines – Part 1: Design requirements. Technical report 61400-1 Ed. 3, International Electrotechnical Commission, 2005 (cited on pages 8, 99, 100, 101, 107, 113, 118, 120, 193).

[86] IEC. Wind turbines – Part 3: Design requirements for offshore wind turbines. Technical report 61400-3 Ed. 1, International Electrotechnical Commission, 2009 (cited on pages 1, 8, 21, 99, 100, 101, 113, 118, 120).

[87] D. Jackson. *Theory of Approximation (Colloquium Publications)*. American Mathematical Society, 1982. ISBN: 978-0-8218-1011-8 (cited on pages 14, 137).

[88] E. N. Jacobs, K. E. Ward, and R. M. Pinkerton. The Characteristics of 78 Related Airfoil Sections from Tests in the Variable-density Wind Tunnel. Technical report 460, NASA, 1933 (cited on page 94).

[89] J. D. Jakeman and A. Narayan. Generation and application of multivariate polynomial quadrature rules. *Computer Methods in Applied Mechanics and Engineering*, **338**:134–161, 2018. DOI: 10.1016/j.cma.2018.04.009 (cited on pages 68, 69, 159).

[90] P. Jantsch, C. G. Webster, and G. Zhang. On the Lebesgue constant of weighted Leja points for Lagrange interpolation on unbounded domains. *IMA Journal of Numerical Analysis*, **39**(2):1039–1057, 2018. DOI: 10.1093/imanum/dry002 (cited on pages 125, 137, 138).

[91] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-MW Reference Wind Turbine for Offshore System Development. Technical report NREL/TP-500-38060, National Renewable Energy Laboratory, 2009. DOI: 10.2172/947422 (cited on pages 101, 104).

[92] V. Kaarnioja. *Smolyak Quadrature*. Master's thesis, University of Helsinki, 2013 (cited on page 22).

[93] D. K. Kahaner and G. Monegato. Nonexistence of extended Gauss–Laguerre and Gauss–Hermite quadrature rules with positive weights. *Zeitschrift für angewandte Mathematik und Physik (ZAMP)*, **29**(6):983–986, 1978. DOI: 10.1007/bf01590820 (cited on page 28).

[94] D. K. Kahaner, J. Waldvogel, and L. W. Fullerton. Addition of points to Gauss–Laguerre quadrature formulas. *SIAM Journal on Scientific and Statistical Computing*, **5**(1):42–55, 1984. DOI: 10.1137/0905003 (cited on page 28).

[95] M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **63**(3):425–464, 2001. DOI: 10.1111/1467-9868.00294 (cited on pages 8, 22, 24, 124, 126, 158, 161, 184).

[96] V. Keshavarzzadeh, R. M. Kirby, and A. Narayan. Numerical integration in multiple dimensions with designed quadrature. *SIAM Journal on Scientific Computing*, **40**(4):A2033–A2061, 2018. DOI: 10.1137/17m1137875 (cited on page 69).

[97] A. D. Kiureghian and O. Ditlevsen. Aleatory or epistemic? Does it matter? *Structural Safety*, **31**(2):105–112, 2009. DOI: 10.1016/j.strusafe.2008.06.020 (cited on page 23).

[98] M. J. Kühn. *Dynamics and design optimisation of offshore wind energy conversion systems*. PhD thesis, Delft University of Technology, 2001. ISBN: 978-90-76468-07-5 (cited on page 100).

[99] P. Kumar. *Multilevel Solvers for Stochastic Fluid Flows*. PhD thesis, Delft University of Technology, 2019. ISBN: 978-94-6366-189-8 (cited on page 11).

[100] S.-D. Kwon. Uncertainty analysis of wind energy potential assessment. *Applied Energy*, **87**(3):856–865, 2010. DOI: 10.1016/j.apenergy.2009.08.038 (cited on page 10).

[101] M. Lackner, A. Rogers, and J. Manwell. Uncertainty analysis in wind resource assessment and wind energy production estimation. In *45th AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics, 2007. DOI: 10.2514/6.2007-1222 (cited on page 10).

[102] D. P. Laurie. Anti-Gaussian quadrature formulas. *Mathematics of Computation*, **65**(214):739–748, 1996. DOI: 10.1090/s0025-5718-96-00713-2 (cited on pages 28, 53).

[103] D. P. Laurie. Calculation of Gauss–Kronrod quadrature rules. *Mathematics of Computation*, **66**(219):1133–1146, 1997. DOI: 10.1090/s0025-5718-97-00861-2 (cited on pages 28, 31, 53, 59).

[104] O. P. Le Maître, O. M. Knio, H. N. Najm, and R. G. Ghanem. Uncertainty propagation using Wiener–Haar expansions. *Journal of Computational Physics*, **197**(1):28–57, 2004. DOI: 10.1016/j.jcp.2003.11.033 (cited on page 10).

[105] O. P. Le Maître and O. M. Knio. *Spectral Methods for Uncertainty Quantification*. Springer, 2010. ISBN: 978-90-481-3519-6. DOI: 10.1007/978-90-481-3520-2 (cited on pages 7, 12, 68, 160).

[106] F. Leja. Sur certaines suites liées aux ensembles plans et leur application à la représentation conforme. *Annales Polonici Mathematici*, **4**(1):8–13, 1957 (cited on pages 46, 47, 61, 68).

[107] J. Li and Y. M. Marzouk. Adaptive construction of surrogates for the Bayesian solution of inverse problems. *SIAM Journal on Scientific Computing*, **36**(3):A1163–A1186, 2014. DOI: 10.1137/130938189 (cited on page 158).

[108] D. Liu, A. Litvinenko, C. Schillings, and V. Schulz. Quantification of airfoil geometry-induced aerodynamic uncertainties—comparison of approaches. *SIAM/ASA Journal on Uncertainty Quantification*, **5**(1):334–352, 2017. DOI: 10.1137/15m1050239 (cited on page 94).

[109] B. A. Lockwood, M. Anitescu, and D. J. Mavriplis. Mixed aleatory/epistemic uncertainty quantification for hypersonic flows via gradient-based optimization and surrogate models. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, 2012. DOI: 10.2514/6.2012-1254 (cited on page 22).

[110] G. J. A. Loeven and H. Bijl. Airfoil analysis with uncertain geometry using the probabilistic collocation method. In *Proceedings of the 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. American Institute of Aeronautics and Astronautics, 2008. DOI: 10.2514/6.2008-2070 (cited on page 94).

[111] J. S. Lopes and M. A. Beaumont. ABC: a useful Bayesian tool for the analysis of population data. *Infection, Genetics and Evolution*, **10**(6):825–832, 2010. DOI: 10.1016/j.meegid.2009.10.010 (cited on page 124).

[112] F. Lu, M. Morzfeld, X. Tu, and A. J. Chorin. Limitations of polynomial chaos expansions in the Bayesian solution of inverse problems. *Journal of Computational Physics*, **282**:138–147, 2015. DOI: 10.1016/j.jcp.2014.11.010 (cited on page 124).

[113] D. S. Lubinsky. A survey of weighted polynomial approximation with exponential weights. *Surveys in Approximation Theory*, **3**:1–105, 2007 (cited on pages 129, 137).

[114] C. Luo, J. Sun, and Y. Wang. Integral estimation from point cloud in d-dimensional space. In *Proceedings of the 25th annual symposium on Computational geometry – SCG '09*. ACM Press, 2009. DOI: 10.1145/1542362.1542389 (cited on pages 160, 168).

[115] J. Ma, V. Rokhlin, and S. Wandzura. Generalized Gaussian quadrature rules for systems of arbitrary functions. *SIAM Journal on Numerical Analysis*, **33**(3):971–996, 1996. DOI: 10.1137/0733048 (cited on page 28).

[116] X. Ma and N. Zabaras. An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. *Journal of Computational Physics*, **228**(8):3084–3113, 2009. DOI: 10.1016/j.jcp.2009.01.006 (cited on pages 100, 101).

[117] X. Ma and N. Zabaras. An efficient Bayesian inference approach to inverse problems based on an adaptive sparse grid collocation method. *Inverse Problems*, **25**(3):035013, 2009. DOI: 10.1088/0266-5611/25/3/035013 (cited on page 124).

[118] N. Macon and A. Spitzbart. Inverses of Vandermonde matrices. *The American Mathematical Monthly*, **65**(2):95–100, 1958. DOI: 10.2307/2308881 (cited on page 50).

[119] J. Mann. Wind field simulation. *Probabilistic Engineering Mechanics*, **13**(4):269–282, 1998. DOI: 10.1016/s0266-8920(97)00036-2 (cited on page 102).

[120] Y. M. Marzouk, H. N. Najm, and L. A. Rahn. Stochastic spectral methods for efficient Bayesian solution of inverse problems. *Journal of Computational Physics*, **224**(2):560–586, 2007. DOI: 10.1016/j.jcp.2006.10.010 (cited on pages 124, 125).

[121] Y. Marzouk and D. Xiu. A stochastic collocation approach to Bayesian inference in inverse problems. *Communications in Computational Physics*, **6**(4):826–847, 2009. DOI: 10.4208/cicp.2009.v6.p826 (cited on pages 124, 140, 148, 158).

[122] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, **21**(6):1087–1092, 1953. DOI: 10.1063/1.1699114 (cited on pages 24, 70, 124, 126, 158).

[123] G. Monegato. A note on extended Gaussian quadrature rules. *Mathematics of Computation*, **30**(136):812–817, 1976. DOI: 10.1090/s0025-5718-1976-0440878-3 (cited on page 28).

[124] G. Monegato. An overview of the computational aspects of Kronrod quadrature rules. *Numerical Algorithms*, **26**(2):173–196, 2001. DOI: 10.1023/a:1016640617732 (cited on page 28).

[125] G. Monegato. Positivity of the weights of extended Gauss–Legendre quadrature rules. *Mathematics of Computation*, **32**(141):243–245, 1978. DOI: 10.1090/s0025-5718-1978-0458809-0 (cited on page 28).

[126] E. C. Morgan, M. Lackner, R. M. Vogel, and L. G. Baise. Probability distributions for offshore wind speeds. *Energy Conversion and Management*, **52**(1):15–26, 2011. DOI: 10.1016/j.enconman.2010.06.015 (cited on page 100).

[127] J. P. Murcia, P. E. Réthoré, A. Natarajan, and J. D. Sørensen. How many model evaluations are required to predict the AEP of a wind power plant? *Journal of Physics: Conference Series*, **625**:012030, 2015. DOI: 10.1088/1742-6596/625/1/012030 (cited on pages 15, 100).

[128] J. P. Murcia, P.-E. Réthoré, N. Dimitrov, A. Natarajan, J. D. Sørensen, P. Graf, and T. Kim. Uncertainty propagation through an aeroelastic wind turbine model using polynomial surrogates. *Renewable Energy*, **119**:910–922, 2018. DOI: 10.1016/j.renene.2017.07.070 (cited on pages 15, 100).

[129] H. N. Najm. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, **41**(1):35–52, 2009. DOI: 10.1146/annurev.fluid.010908.165248 (cited on pages 7, 12, 68, 100).

[130] A. Narayan and J. D. Jakeman. Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation. *SIAM Journal on Scientific Computing*, **36**(6):A2952–A2983, 2014. DOI: 10.1137/140966368 (cited on pages 47, 68, 125, 128, 129, 159).

[131] M. Navarro, J. Witteveen, and J. Blom. Stochastic collocation for correlated inputs. In M. Papadrakis, V. Papadopoulos, and G. Stefanou, editors, *Proceedings of the 1st International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2015)*, pages 218–236. Institute of Structural Analysis and Antiseismic Research School of Civil Engineering National Technical University of Athens (NTUA) Greece, 2015. DOI: 10.7712/120215.4266.544 (cited on page 68).

[132] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial & Applied Mathematics (SIAM), 1992. ISBN: 978-0-89871-295-7. DOI: 10.1137/1.9781611970081 (cited on pages 100, 176).

[133] F. Nobile, L. Tamellini, and R. Tempone. Comparison of Clenshaw–Curtis and Leja quasi-optimal sparse grids for the approximation of random PDEs. In *Lecture Notes in Computational Science and Engineering*, pages 475–482. Springer International Publishing, 2015. DOI: 10.1007/978-3-319-19800-2_44 (cited on page 47).

[134] E. Novak and K. Ritter. High dimensional integration of smooth functions over cubes. *Numerische Mathematik*, **75**(1):79–97, 1996. DOI: 10.1007/s002110050231 (cited on pages 21, 128).

[135] E. Novak and K. Ritter. Simple cubature formulas with high polynomial exactness. *Constructive Approximation*, **15**(4):499–522, 1999. DOI: 10.1007/s003659900119 (cited on pages 21, 22, 36, 68, 89, 109, 159).

[136] A. O'Hagan and J. E. Oakley. Probability is perfect, but we can't elicit it perfectly. *Reliability Engineering & System Safety*, **85**(1-3):239–248, 2004. DOI: 10.1016/j.ress.2004.03.014 (cited on page 23).

[137] S. Oladyshkin and W. Nowak. Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion. *Reliability Engineering & System Safety*, **106**:179–190, 2012. DOI: 10.1016/j.ress.2012.05.002 (cited on page 69).

[138] T. A. Oliver and R. D. Moser. Bayesian uncertainty quantification applied to RANS turbulence models. *Journal of Physics: Conference Series*, **318**(4):042032, 2011. DOI: 10.1088/1742-6596/318/4/042032 (cited on page 124).

[139] F. Palacios, T. D. Economon, A. Aranake, S. R. Copeland, A. K. Lonkar, T. W. Lukaczyk, D. E. Manosalvas, K. R. Naik, S. Padron, B. Tracey, A. Variyar, and J. J. Alonso. Stanford university unstructured (SU2): analysis and design technology for turbulent flows. In *Proceedings of the 52nd Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 2014. DOI: 10.2514/6.2014-0243 (cited on pages 94, 151, 184).

[140] E. Passow. Another proof of Jackson's theorem. *Journal of Approximation Theory*, **3**(2):146–148, 1970. DOI: 10.1016/0021-9045(70)90022-5 (cited on pages 14, 137).

[141] T. N. L. Patterson. An algorithm for generating interpolatory quadrature rules of the highest degree of precision with preassigned nodes for general weight functions. *ACM Transactions on Mathematical Software*, **15**(2):123–136, 1989. DOI: 10.1145/63522.63523 (cited on pages 45, 53).

[142] T. N. L. Patterson. The optimum addition of points to quadrature formulae. *Mathematics of Computation*, **22**(104):847–856, 1968. DOI: 10.1090/s0025-5718-68-99866-9 (cited on pages 28, 31, 45).

[143] B. Peherstorfer and Y. Marzouk. A transport-based multifidelity preconditioner for Markov chain Monte Carlo. *Advances in Computational Mathematics*:1–28, 2019. DOI: 10.1007/s10444-019-09711-y (cited on page 158).

[144] F. Peherstorfer. Characterization of positive quadrature formulas. *SIAM Journal on Mathematical Analysis*, **12**(6):935–942, 1981. DOI: 10.1137/0512079 (cited on page 28).

[145] G. Petrone, C. de Nicola, D. Quagliarella, J. Witteveen, and G. Iaccarino. Wind turbine performance analysis under uncertainty. In *Proceedings of the 49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, 2011. DOI: 10.2514/6.2011-544 (cited on page 15).

[146] F. Piazzon, A. Sommariva, and M. Vianello. Caratheodory–Tchakaloff subsampling. *Dolomites Research Notes on Approximation*, **10**:5–14, 2017. DOI: 10.14658/pupj-drna-2017-1-2 (cited on page 69).

[147] M. Pisaroni, F. Nobile, and P. Leyland. Continuation Multi-Level Monte-Carlo method for Uncertainty Quantification in Turbulent Compressible Aerodynamics Problems modeled by RANS. Technical report 10.2017, École Polytechnique Fédérale de Lausanne, 2017. DOI: 10.5075/epfl-mathicse-228214 (cited on page 186).

[148] G. Pólya. Über die konvergenz von quadraturverfahren. *Mathematische Zeitschrift*, **37**(1):264–286, 1933. DOI: 10.1007/bf01474574 (cited on page 30).

[149] S. Prudhomme and C. M. Bryant. Adaptive surrogate modeling for response surface approximations with application to Bayesian inference. *Advanced Modeling and Simulation in Engineering Sciences*, **2**(1), 2015. DOI: 10.1186/s40323-015-0045-5 (cited on page 124).

[150] P. Rabinowitz, J. Kautsky, S. Elhay, and J. Butcher. On sequences of imbedded integration rules. Technical report 86-02, The University of Adelaide, 1986 (cited on page 32).

[151] P. Ragan and L. Manuel. Statistical extrapolation methods for estimating wind turbine extreme loads. In *Proceedings of the 45th AIAA Aerospace Sciences Meeting and Exhibit*. American Institute of Aeronautics and Astronautics, 2007. DOI: 10.2514/6.2007-1221 (cited on page 100).

[152] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006. ISBN: 978-0-262-18253-9 (cited on pages 10, 168).

[153] J. Ray, S. Lefantzi, S. Arunajatesan, and L. J. DeChant. Bayesian calibration of a RANS model with a complex response surface - a case study with jet-in-crossflow configuration. In *Proceedings of the 45th AIAA Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, 2015. DOI: 10.2514/6.2015-2784 (cited on page 124).

[154] J. M. Rinker. Calculating the sensitivity of wind turbine loads to wind inputs using response surfaces. *Journal of Physics: Conference Series*, **753**:032057, 2016. DOI: 10.1088/1742-6596/753/3/032057 (cited on page 15).

[155] A. A. Rushdi, L. P. Swiler, E. T. Phipps, M. D'Elia, and M. S. Ebeida. VPS: voronoi piecewise surrogate models for high-dimensional data fitting. *International Journal for Uncertainty Quantification*, **7**(1):1–21, 2017. DOI: 10.1615/int.j.uncertaintyquantification.2016018697 (cited on page 168).

[156] E. K. Ryu and S. P. Boyd. Extensions of Gauss quadrature via linear programming. *Foundations of Computational Mathematics*, **15**(4):953–971, 2014. DOI: 10.1007/s10208-014-9197-9 (cited on page 69).

[157] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer New York, 2018. ISBN: 978-1-4939-8845-7. DOI: 10.1007/978-1-4939-8847-1 (cited on page 22).

[158] K. Sargsyan, H. N. Najm, and R. Ghanem. On the statistical calibration of physical models. *International Journal of Chemical Kinetics*, **47**(4):246–276, 2015. DOI: 10.1002/kin.20906 (cited on page 124).

[159] C. Schillings and C. Schwab. Sparse, adaptive Smolyak quadratures for Bayesian inverse problems. *Inverse Problems*, **29**(6):065011, 2013. DOI: 10.1088/0266-5611/29/6/065011 (cited on pages 68, 158, 183).

[160] L. Schröder, N. K. Dimitrov, D. R. Verelst, and J. A. Sørensen. Wind turbine site-specific load estimation using artificial neural networks calibrated by means of high-fidelity load simulations. *Journal of Physics: Conference Series*, **1037**:062027, 2018. DOI: 10.1088/1742-6596/1037/6/062027 (cited on page 100).

[161] C. Schwab and A. M. Stuart. Sparse deterministic approximation of Bayesian inverse problems. *Inverse Problems*, **28**(4):045003, 2012. DOI: 10.1088/0266-5611/28/4/045003 (cited on page 158).

[162] P. Seshadri, A. Narayan, and S. Mahadevan. Effectively subsampled quadratures for least squares polynomial approximations. *SIAM/ASA Journal on Uncertainty Quantification*, **5**(1):1003–1023, 2017. DOI: 10.1137/16m1057668 (cited on page 69).

[163] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976. ISBN: 978-0-691-10042-5 (cited on page 22).

[164] H. Shah, S. Hosder, and T. Winter. A mixed uncertainty quantification approach using evidence theory and stochastic expansions. *International Journal for Uncertainty Quantification*, **5**(1):21–48, 2015. DOI: 10.1615/int.j.uncertaintyquantification.2015010941 (cited on page 22).

[165] Y. Shin and D. Xiu. A randomized algorithm for multivariate function approximation. *SIAM Journal on Scientific Computing*, **39**(3):A983–A1002, 2017. DOI: 10.1137/16m1075193 (cited on page 69).

[166] J. A. Shohat and J. D. Tamarkin. *The problem of moments*, number 1 in Mathematical Surveys. American Mathematical Society, fourth edition, 1943. ISBN: 978-0-8218-1501-4 (cited on page 65).

[167] M. Sinsbeck and W. Nowak. An optimal sampling rule for nonintrusive polynomial chaos expansions of expensive models. *International Journal for Uncertainty Quantification*, **5**(3):275–295, 2015. DOI: 10.1615/int.j.uncertaintyquantification.2015008446 (cited on page 68).

[168] J. W. Slater, J. C. Dudek, and K. E. Tatum. The NPARC alliance verification and validation archive. In *Proceedings of the ASME Fluids Engineering Division Summer Meeting*, volume 1, pages 1005–1014. ASME, 2000 (cited on page 150).

[169] S. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Proceedings of the USSR Academy of Sciences*, **148**(5):1042–1045, 1963 (cited on pages 21, 68, 109, 159).

[170] P. R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. In *Proceedings of the 30th Aerospace Sciences Meeting and Exhibit*, number 92-0439, pages 6–9. American Institute of Aeronautics and Astronautics, 1992. DOI: 10.2514/6.1992-439 (cited on pages 151, 184).

[171] L. E. S. Stieng, R. Hetland, S. Schafhirt, and M. Muskulus. Relative assessment of fatigue loads for offshore wind turbine support structures. *Energy Procedia*, **80**:229–236, 2015. DOI: 10.1016/j.egypro.2015.11.426 (cited on page 100).

[172] L. E. S. Stieng and M. Muskulus. Reducing the number of load cases for fatigue damage assessment of offshore wind turbine support structures using a simple severity-based sampling method. *Wind Energy Science*, **3**(2):805–818, 2018. DOI: 10.5194/wes-3-805-2018 (cited on page 100).

[173] A. M. Stuart and A. L. Teckentrup. Posterior consistency for Gaussian process approximations of Bayesian posterior distributions. *Mathematics of Computation*, **87**(310):721–753, 2017. DOI: 10.1090/mcom/3244 (cited on pages 124, 158).

[174] T. J. Sullivan. *Introduction to Uncertainty Quantification*. Springer International Publishing, 2015. ISBN: 978-3-319-23394-9. DOI: 10.1007/978-3-319-23395-6 (cited on pages 7, 160).

[175] B. Sündermann. Lebesgue constants in Lagrangrian interpolation at the Fekete points. Technical report 44, Universität Dortmund, Lehrstuhl Mathematik III, 1980 (cited on page 14).

[176] L. P. Swiler, T. L. Paez, and R. L. Mayes. Epistemic uncertainty quantification tutorial. In *Proceedings of the IMAC-XXVII*. Society for Experimental Mechanics, 2009 (cited on page 22).

[177] R. Taylor. *Lagrange Interpolation on Leja points*. PhD thesis, University of South Florida, 2008 (cited on page 138).

[178] R. Taylor and V. Totik. Lebesgue constants for Leja points. *IMA Journal of Numerical Analysis*, **30**(2):462–486, 2010. DOI: 10.1093/imanum/drn082 (cited on pages 47, 138).

[179] S. T. Tokdar and R. E. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, **2**(1):54–60, 2009. DOI: 10.1002/wics.56 (cited on page 158).

[180] L. N. Trefethen. *Spectral Methods in MATLAB*. Society for Industrial & Applied Mathematics (SIAM), 2000. ISBN: 978-0-89871-465-4. DOI: 10.1137/1.9780898719598 (cited on pages 12, 19, 28).

[181] R. K. Tripathy and I. Bilionis. Deep UQ: learning deep neural network surrogate models for high dimensional uncertainty quantification. *Journal of Computational Physics*, **375**:565–588, 2018. DOI: 10.1016/j.jcp.2018.08.036 (cited on page 10).

[182] K. L. Van Buren, M. G. Mollineaux, F. M. Hemez, and S. Atamturktur. Simulating the dynamics of wind turbine blades: part II, model validation and uncertainty quantification. *Wind Energy*, **16**(5):741–758, 2012. DOI: 10.1002/we.1522 (cited on page 24).

[183] B. Vandewoestyne and R. Cools. Good permutations for deterministic scrambled Halton sequences in terms of L_2-discrepancy. *Journal of Computational and Applied Mathematics*, **189**(1-2):341–361, 2006. DOI: 10.1016/j.cam.2005.05.022 (cited on page 11).

[184] P. S. Veers. Three-dimensional wind simulation. Technical report SAND88–0152, Sandia National Laboratories, 1988 (cited on page 102).

[185] D. Vladislav. Construction of Gauss–Kronrod–Hermite quadrature and cubature formulas. *Studia Universitatis Babeş-Bolyai Mathematica*, **49**(3):111–117, 2004 (cited on page 28).

[186] J. A. Vrugt, C. J. F. ter Braak, C. G. H. Diks, B. A. Robinson, J. M. Hyman, and D. Higdon. Accelerating Markov chain Monte Carlo simulation by differential evolution with self-adaptive randomized subspace sampling. *International Journal of Nonlinear Sciences and Numerical Simulation*, **10**(3):273–290, 2009. DOI: 10.1515/ijnsns.2009.10.3.273 (cited on pages 124, 158).

[187] P. Wacker. Laplace's method in Bayesian inverse problems, 2017. arXiv: 1701.07989v2 [math.PR] (cited on page 158).

[188] G. A. Watson. *Approximation Theory and Numerical Methods*. John Wiley & Sons Ltd., 1980. ISBN: 978-0-471-27706-4 (cited on pages 12, 14, 30, 62).

[189] K. Weichselberger. The theory of interval-probability as a unifying concept for uncertainty. *International Journal of Approximate Reasoning*, **24**(2-3):149–170, 2000. DOI: 10.1016/s0888-613x(00)00032-3 (cited on page 22).

[190] E. J. Werkhoeven and J. P. Verhoef. Offshore Meteorological Mast IJmuiden: Abstract of Intrumentation Report. Technical report ECN-Wind Memo-12-010, Energieonderzoek Centrum Nederland (ECN), 2012 (cited on pages 101, 103).

[191] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, **60**(4):897–936, 1938. DOI: 10.2307/2371268 (cited on page 12).

[192] D. C. Wilcox. *Turbulence modeling for CFD*. DCW industries La Cañada, CA, third edition, 2006. ISBN: 978-1-928729-08-2 (cited on pages 150, 183).

[193] D. J. Wilkinson and S. K. H. Yeung. Adaptive Metropolis–Hastings samplers for the Bayesian analysis of large linear Gaussian systems. *Computing Science and Statistics*, **33**:128–138, 2002 (cited on page 124).

[194] M. W. Wilson. A general algorithm for nonnegative quadrature formulas. *Mathematics of Computation*, **23**(106):253–258, 1969. DOI: 10.1090/s0025-5718-1969-0242374-1 (cited on pages 32, 69).

[195] J. A. S. Witteveen, A. Doostan, R. Pecnik, and G. Iaccarino. Uncertainty quantification of the transonic flow around the RAE 2822 airfoil. In *Annual Research Briefs 2009, Stanford University Center for Turbulence Research*, pages 93–104. 2009 (cited on pages 94, 95, 186).

[196] J. A. S. Witteveen and G. Iaccarino. Simplex elements stochastic collocation for uncertainty propagation in robust design optimization. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, 2010. DOI: 10.2514/6.2010-1313 (cited on page 168).

[197] J. A. S. Witteveen and G. Iaccarino. Simplex stochastic collocation with ENO-type stencil selection for robust uncertainty quantification. *Journal of Computational Physics*, **239**:1–21, 2013. DOI: 10.1016/j.jcp.2012.12.030 (cited on page 168).

[198] J. A. S. Witteveen and G. Iaccarino. Simplex stochastic collocation with random sampling and extrapolation for nonhypercube probability spaces. *SIAM Journal on Scientific Computing*, **34**(2):A814–A838, 2012. DOI: 10.1137/100817504 (cited on page 168).

[199] K. Wu, Y. Shin, and D. Xiu. A randomized tensor quadrature method for high dimensional polynomial approximation. *SIAM Journal on Scientific Computing*, **39**(5):A1811–A1833, 2017. DOI: 10.1137/16m1081695 (cited on page 69).

[200] D. Xiu. Fast numerical methods for stochastic computations: a review. *Communications in Computational Physics*, **5**(2–4):242–272, 2009 (cited on page 12).

[201] D. Xiu. *Numerical Methods for Stochastic Computations*. Princeton University Press, 2010. ISBN: 978-0-691-14212-8 (cited on pages 7, 12, 22, 68, 100, 124, 160).

[202] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, **27**(3):1118–1139, 2005. DOI: 10.1137/040615201 (cited on page 12).

[203] L. Yan and Y.-X. Zhang. Convergence analysis of surrogate-based methods for Bayesian inverse problems. *Inverse Problems*, **33**(12):125001, 2017. DOI: 10.1088/1361-6420/aa9417 (cited on page 140).

[204] L. Yan and T. Zhou. Adaptive multi-fidelity polynomial chaos approach to Bayesian inference in inverse problems. *Journal of Computational Physics*, **381**:110–128, 2019. DOI: 10.1016/j.jcp.2018.12.025 (cited on page 124).

[205] L. Zeng, L. Shi, D. Zhang, and L. Wu. A sparse grid based Bayesian method for contaminant source identification. *Advances in Water Resources*, **37**:1–9, 2012. DOI: 10.1016/j.advwatres.2011.09.011 (cited on page 124).

[206] G. Zhang, D. Lu, M. Ye, M. Gunzburger, and C. Webster. An adaptive sparse-grid high-order stochastic collocation method for Bayesian inference in groundwater reactive transport modeling. *Water Resources Research*, **49**(10):6871–6892, 2013. DOI: 10.1002/wrcr.20467 (cited on pages 24, 68, 124).

Index

Summary

Two sources of uncertainty can be distinguished in models for wind turbine load calculations. Firstly, the environment that the turbine has to withstand is inherently uncertain and it directly influences the power output and life time of a wind turbine. Secondly, the models used to predict the forces acting on a wind turbine form a source of uncertainty, since these models contain possibly unknown assumptions, biases, and simplifications. Accurately predicting these forces is crucial, since these directly affect the power output and life time.

Both types of uncertainties can be modeled using probability theory. The environmental conditions are often provided as a distribution (in fact, they are standardized as such) or as a set of measurements and both can be embedded straightforwardly in probability theory. The uncertainty arising from assumptions, biases, and simplifications can be assessed using Bayesian model calibration, yielding a probability distribution of the model parameters. This requires measurement data of an output of the model. If all relevant distributions are characterized, predictions that incorporate uncertainty can be inferred using Monte Carlo methods.

However, these methods often converge prohibitively slow, since doubling the accuracy of the estimate requires quadrupling the number of model evaluations. These model evaluations are computationally costly, since each evaluation consists of the complex proces of estimating the forces acting on a turbine.

The goal of this thesis is to numerically assess the effect of both external and model uncertainty using stochastic collocation techniques and apply these techniques to wind turbine load calculations. All methods considered in this thesis are by design non-intrusive, meaning that all methods are based on exploiting only general characteristics of the model (such as its smoothness properties) and use only a finite number of evaluations of the model.

The methods in this thesis are based on polynomial approximation. In essence the computationally costly model is replaced by a polynomial, which is cheap to represent and evaluate using a computer. If the polynomial is constructed well, high convergence rates are obtained for sufficiently smooth functions. The main focus is on numerical

integration using quadrature rules, since statistical moments such as the mean and standard deviation are integral quantities. In addition, polynomial interpolation is considered, since that yields a computationally cheap approximation that can fully replace the model.

The problem of numerical integration is cast as constructing an efficient quadrature rule, which consists of a weighted sum of a finite number of model evaluations. Three properties are relevant for a quadrature rule:

1. that the rule has positive weights,
2. that the rule integrates all functions from a high-dimensional polynomial space exactly, and
3. that the rule is nested (i.e. the nodes of smaller rules are reused by larger rules).

If a rule has positive weights and integrates a large number of polynomials, it is numerically stable and converges under mild assumptions.

To facilitate the construction of quadrature rules, a framework is proposed that describes the addition, replacement, and removal of nodes in a quadrature rule setting. All modifications preserve positive weights and keep a quadrature rule interpolatory, meaning that its number of nodes equals the number of polynomials it integrates exactly. Therefore the addition and removal of nodes can be used to construct sequences of interpolatory quadrature rules with positive weights. However, it is demonstrated that it is not always possible to add nodes to a quadrature rule such that positive weights are preserved.

By initializing the framework with random samples, a quadrature rule with positive weights is derived that can accurately compute integrals with respect to distributions described by sample sets. If these rules are used to compute integrals of polynomials, the same estimate is obtained as if the polynomials are integrated using the (large) set of samples. Constructing a quadrature rule using samples does not require any knowledge about the distribution, domain of definition, or dimension of the space beforehand. This quadrature rule is called the *implicit quadrature* rule in this thesis.

The implicit quadrature rule is very suitable for computing the equivalent loads acting on an offshore wind turbine, since measurements conducted at an offshore site can be incorporated accurately and efficiently. To demonstrate this, the NREL 5MW benchmark wind turbine and freely available offshore measurements conducted at the North Sea are employed. The computed equivalent loads are of the same order of magnitude as the standardized approach (which is “binning”). However, to use the proposed procedure significantly less computational effort is necessary.

Based on the approaches proposed for numerically propagating input uncertainty, two methods are discussed to statistically assess the effect of model uncertainty. This type of uncertainty is modeled using a Bayesian framework in this thesis. It is challenging to statistically assess this uncertainty, since the obtained distribution (the *posterior*) depends explicitly on the computationally expensive model.

Firstly polynomial interpolation is considered. By replacing the model with a polynomial, it is feasible to assess the posterior distribution straightforwardly using Monte Carlo methods. For this purpose a new nodal set based on Leja nodes is proposed, which

adaptively incorporates an existing approximation of the posterior. This approach is very suitable if the model under consideration is smooth or analytic, which is demonstrated by calibrating the turbulence closure coefficients of the Spalart–Allmaras turbulence model.

Secondly the implicit quadrature rule is considered, which is modified to accommodate the posterior distribution. Since a quadrature rule is very suitable for computing integral quantities, this approach can accurately compute integrals weighted with the posterior, which are often obtained when inferring Bayesian predictions. The posterior distribution is incorporated by interpolating available point evaluations of the posterior and using this interpolant to refine the quadrature rule. The obtained rules have positive weights, so these rules are numerically stable and can be used to accurately compute integrals of a broad class of integrands. However, if the model is smooth, the convergence rate is smaller than the rate of polynomial interpolation approaches. Again, the turbulence closure coefficients of the Spalart–Allmaras turbulence model are considered and Bayesian predictions are inferred using a statistical model incorporating among others measurement and model error.

Numerical examples are discussed throughout this thesis to demonstrate the applicability of the proposed approaches. A common theme in all results is that the quadrature rules constructed in this thesis are based on arbitrarily large sample sets. Thanks to this, the computational time of constructing the quadrature rule is independent of the number of parameters, the polynomial basis under consideration, or the distribution of the sample set (if there exists one). This results in a set of highly flexible quadrature rules that provide rapidly converging estimates for smooth integrands, whereas numerical stability is guaranteed by ensuring positive weights.

Samenvatting

Bij het modelleren van offshore windenergie kan er onderscheid worden gemaakt tussen twee soorten onzekerheden. Als eerste is de omgeving van de windturbine veranderlijk en zit er een inherente onzekerheid in de mate waarin zij de energieproductie en levensduur van de turbine beïnvloedt. Ten tweede bevat het model waarmee de krachten op de turbine worden berekend, onzekerheden als gevolg van de intrinsieke (soms zelfs onbekende) aannames, afwijkingen en vereenvoudigingen van het model. Het is cruciaal deze krachten accuraat te berekenen, omdat deze een directe invloed hebben op de energieopbrengst en levensduur van de turbine.

Deze twee soorten onzekerheden kunnen allebei gemodelleerd worden met behulp van kansrekening. De onzekerheid in de omgeving wordt gewoonlijk beschreven op een gestandaardiseerde manier met een kansverdeling of door middel van metingen. Beide gevallen kunnen eenvoudig beschreven worden met kansrekening. De onzekerheid als gevolg van de imperfecties van het model wordt geïdentificeerd met behulp van Bayesiaanse kalibratie, wat ook een kansdichtheid oplevert. Dit vereist wel meetdata van de gewenste uitvoer van het model. Zodra alle onzekerheden door middel van kansdichtheden beschreven zijn, kunnen voorspellingen worden gegenereerd door middel van Monte-Carlosimulatie.

Monte-Carlosimulatie convergeert echter bijzonder langzaam. Om de nauwkeurigheid te verdubbelen zijn vier maal zo veel evaluaties van het model nodig. Deze evaluaties kosten veel rekenkracht, omdat steeds weer de krachten die op de turbine werken herberekend moeten worden.

Het doel van deze dissertatie is het effect van zowel de onzekere omgeving als de onzekere modelimperfecties numeriek te bestuderen. Hiervoor wordt gebruik gemaakt van collocatie-technieken, waarmee de krachten op een windturbine worden uitgerekend. Alle technieken die in deze dissertatie aan de orde komen, benutten alleen de algemene eigenschappen van het model, waardoor dat niet aangepast hoeft te worden. Oftewel de methoden zijn “non-intrusive” en vereisen dus alleen een eindig aantal evaluaties van het model.

Alle besproken methoden zijn gebaseerd op polynomiale benadering. In de kern

bestaat deze aanpak uit het vervangen van het model door een polynoom, dat goed gerepresenteerd en snel geëvalueerd kan worden door middel van een computer. Als het model glad is, kan snelle convergentie verkregen worden, mits het polynoom zorgvuldig geconstrueerd is. De nadruk ligt voornamelijk op numerieke integratie met behulp van kwadratuurregels, aangezien de momenten van een kansverdeling (zoals het gemiddelde en de standaarddeviatie) integralen zijn. Polynomiale interpolatie wordt daarbij ook in ogenschouw genomen, aangezien dat een benadering oplevert welke het model volledig kan vervangen.

Numerieke integratie wordt in deze dissertatie omgeschreven als het efficiënt construeren van kwadratuurregels, welke bestaan uit een gewogen som van een eindig aantal evaluaties van het model. Drie eigenschappen zijn van belang bij een kwadratuurregel:

1. dat de regel positieve gewichten heeft,
2. dat de regel alle functies uit een hoog-dimensionale polynomiale ruimte exact integreert en
3. dat de regel genest is (oftewel kleinere kwadratuurregels gebruiken een deelverzameling van de model evaluaties van een grotere kwadratuurregel).

Als de regel positieve gewichten heeft en bovendien een groot aantal polynomen exact integreert, dan convergeert deze voor een groot aantal functies en is deze numeriek stabiel.

Voor het construeren van kwadratuurregels wordt een wiskundig kader gepresenteerd waarmee punten kunnen worden toegevoegd aan, vervangen in of verwijderd uit een kwadratuurregel. Ondanks de modificaties behoudt de regel de positieve gewichten en de eigenschap dat het aantal punten gelijk is aan het aantal polynomen dat de regel exact integreert. Gebruik makende van het kader wordt aangetoond dat het niet altijd mogelijk is punten aan een kwadratuurregel toe te voegen op deze wijze.

Door het kader echter te initialiseren met samples van een kansdichtheid wordt een verzameling kwadratuurregels verkregen die positieve gewichten hebben en bovendien accuraat integralen kunnen uitrekenen. Wanneer die regels gebruikt worden om integralen over polynomen uit te rekenen, wordt eenzelfde schatting verkregen als wanneer de integralen worden uitgerekend met de aanvankelijke samples. Het op deze manier construeren van kwadratuurregels vereist nagenoeg geen kennis van de kansdichtheid van de samples, het domein van de kansdichtheid of de dimensie van het domein. Deze kwadratuurregel wordt derhalve de *impliciete kwadratuurregel* genoemd in deze dissertatie.

De impliciete kwadratuurregel leent zich bijzonder goed voor het berekenen van equivalente belastingen op een offshore windturbine, omdat metingen accuraat en efficiënt kunnen worden meegenomen. Dit wordt gedemonstreerd door de NREL 5MW windturbine te simuleren, gebruik makende van metingen van de weersomstandigheden op de Noordzee. De berekende equivalente belastingen hebben dezelfde orde van grootte als de belastingen berekend met de gestandaardiseerde aanpak (dat is “binnen”). Voor de gepostuleerde technieken zijn echter significant minder model berekeningen vereist.

Er worden ook twee methoden besproken om onzekerheid in het model te karakteriseren. Beide methoden zijn gebaseerd op aanpakken waarmee onzekerheid gepropageerd kan worden. De onzekerheid van het model wordt in deze dissertatie beschreven met een Bayesiaanse analyse. Het is verre van eenvoudig om deze onzekerheid statistisch te kwantificeren, aangezien de verkregen distributie (de *posterior*) expliciet afhangt van het model en het model evalueren veel rekenkracht vereist.

Als eerste wordt polynomiale interpolatie beschouwd. Door het model te vervangen door een polynoom wordt het mogelijk om statistische inferentie te doen door middel van Monte-Carlosimulatie. Specifiek hiervoor wordt een nieuwe verzameling van collocatie-punten afgeleid, welke gebaseerd is op Leja punten en adaptief een bestaande benadering van de posterior meeneemt. Deze aanpak is in het bijzonder geschikt wanneer het model glad of analytisch is, hetgeen gedemonstreerd wordt door de parameters van het Spalart–Allmaras turbulentie model te kalibreren.

Ten tweede wordt de impliciete kwadratuurregel opnieuw benut, maar nu in een aangepaste vorm om de posterior te kunnen gebruiken. Aangezien een kwadratuurregel bijzonder geschikt is voor het berekenen van integralen, is het mogelijk accuraat statistische momenten van de posterior uit te rekenen, welke vaak verkregen worden bij inferentie van Bayesiaanse voorspellingen. De posterior wordt meegenomen door beschikbare evaluaties van deze kansdichtheid te interpoleren en de verkregen benadering te gebruiken bij het construeren van de impliciete kwadratuurregel. De verkregen regel heeft positieve gewichten en is daardoor stabiel en accuraat voor een grote functieklasse. Als het model glad is, convergeert deze aanpak langzamer wanneer deze vergeleken wordt met polynomiale interpolatie. Opnieuw worden de parameters van het Spalart–Allmaras turbulentie model beschouwd en Bayesianse voorspellingen worden verkregen gebruik makende van een statistisch model dat zowel modelonzekerheid als meetfouten bevat.

De toepasbaarheid van de nieuwe methoden uit deze dissertatie wordt gedemonstreerd door middel van vele numerieke voorbeelden. Een regelmatig terugkerend thema is dat de kwadratuurregels zijn gebaseerd op samples. Dankzij dit is de rekentijd van het construeren van de kwadratuurregel onafhankelijk van het aantal onzekere parameters, de polynomiale basis of de distributie van de samples (mits deze bestaat). Het resultaat bestaat uit zeer flexibele kwadratuurregels, welke numeriek stabiel zijn en waarmee snel convergerende benaderingen van integralen verkregen kunnen worden.

Curriculum Vitae

Laurent van den Bos

17-02-1992	Born in Heerlen, the Netherlands

Education

	Grotius College, Heerlen
2004–2010	Voorbereidend Wetenschappelijk Onderwijs
	Eindhoven University of Technology
2010–2011	Propedeuse Computer Science and Engineering
2010–2011	Propedeuse Industrial and Applied Mathematics
2010–2013	Bachelor Industrial and Applied Mathematics
2013–2015	Master Industrial and Applied Mathematics (cum laude)
	Centrum Wiskunde & Informatica, **Delft University of Technology**
2016–2020	PhD Candidate

List of publications

- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes. *Communications in Computational Physics*, **27**(1):33–69, 2020. DOI: 10.4208/cicp.oa-2018-0218. arXiv: 1802.02035 [math.NA].

- L. M. M. van den Bos. The implicit quadrature rule. Zenodo, Software, 2019. DOI: 10.5281/zenodo.3234434.

- L. M. M. van den Bos, B. Sanderse, and W. A. A. M. Bierbooms. Adaptive sampling-based quadrature rules for efficient Bayesian prediction. *Under review*, 2019. arXiv: 1907.08418 [math.NA].

- L. M. M. van den Bos, W. A. A. M. Bierbooms, A. Alexandre, B. Sanderse, and G. J. W. van Bussel. Fatigue design load calculations of the offshore NREL 5MW benchmark turbine using quadrature rule techniques. *To appear in Wind Energy*, 2019. arXiv: 1904.07021 [cs.CE].

- L. M. M. van den Bos and B. Sanderse. A geometric approach for the addition of nodes to an interpolatory quadrature rule with positive weights. *Under review*, 2019. arXiv: 1902.07477 [math.NA].

- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Generating nested quadrature rules with positive weights based on arbitrary sample sets. *To appear in SIAM/ASA Journal on Uncertainty Quantification*, 2018. arXiv: 1809.09842 [math.NA].

- L. M. M. van den Bos, B. Sanderse, L. Blonk, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Efficient ultimate load estimation for offshore wind turbines using interpolating surrogate models. *Journal of Physics: Conference Series*, **1037**:062017, 2018. DOI: 10.1088/1742-6596/1037/6/062017.

- L. M. M. van den Bos and B. Sanderse. Uncertainty quantification for wind energy applications. Technical report SC-1701, Centrum Wiskunde & Informatica, 2017.

- L. M. M. van den Bos, B. Koren, and R. P. Dwight. Non-intrusive uncertainty quantification using reduced cubature rules. *Journal of Computational Physics*, **332**:418–445, 2017. DOI: 10.1016/j.jcp.2016.12.011. arXiv: 1905.06177 [math.NA].

- L. M. M. van den Bos, B. Koren, and R. P. Dwight. Improved non-intrusive uncertainty propagation in complex fluid flow problems. In *Proceedings of the 9th International Conference on Computational Fluid Dynamics (ICCFD9)*, number ICCFD9-2016-213, 2016.

List of attended conferences with presentation

- L. M. M. van den Bos and B. Sanderse. Iterative construction of quadrature rules for Bayesian prediction. *Frontiers of Uncertainty Quantification in Fluid Dynamics (FrontUQ)*, Pisa, Italy, 2019.

- L. M. M. van den Bos and B. Sanderse. Sampling-based quadrature rules for forward and inverse problems. *3rd International Conference on Uncertainty Quantification in Computational Science and Engineering (UNCECOMP)*, Hersonissos (Crete), Greece, 2019.

- L. M. M. van den Bos and B. Sanderse. Constructing positive and nested quadrature rules for Bayesian prediction. *SIAM Conference on Computational Science and Engineering (SIAM CSE)*, Spokane (Washington), United States of America, 2019.

- L. M. M. van den Bos. Efficient load case calculations by means of collocation approaches. *BLADED User Conference*, Hamburg, Germany, 2018.
- L. M. M. van den Bos and B. Sanderse. The implicit quadrature rule for uncertainty quantification. *43rd Woudschoten Conference*, Zeist, the Netherlands, 2018.
- L. M. M. van den Bos. Constructing quadrature rules using samples. *5th Workshop on Sparse Grids and Applications (SGA)*, Munich, Germany, 2018.
- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Efficient ultimate load estimation for offshore wind turbines. *The Science of Making Torque from Wind (TORQUE)*, Milano, Italy, 2018.
- L. M. M. van den Bos and B. Sanderse. Collocation in Bayesian calibration problems. *SIAM Conference on Uncertainty Quantification (SIAM UQ)*, Garden Grove (California), United States of America, 2018.
- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian calibration of turbulence models using an interpolating surrogate. *Computational Sciences for Future Energy Conference (CSER)*, Eindhoven, the Netherlands, 2017.
- L. M. M. van den Bos and B. Sanderse. Bayesian calibration of turbulence models using an interpolating surrogate. *Frontiers of Uncertainty Quantification in Engineering (FrontUQ)*, Munich, Germany, 2017.
- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian calibration of model-form uncertainty applied to wind turbine wake simulation. *SIAM Conference on Computational Science and Engineering (SIAM CSE)*, Atlanta (Georgia), United States of America, 2017.
- L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel. Bayesian calibration of model uncertainty. *41st Woudschoten Conference*, Zeist, the Netherlands, 2016.
- L. M. M. van den Bos, B. Koren, and R. P. Dwight. Improved non-intrusive uncertainty propagation. *9th International Conference on Computational Fluid Dynamics (ICCFD9)*, Istanbul, Turkey, 2016.

Acknowledgments

As I mentioned in the preface of this thesis, it is difficult to describe the amount of fun I had during the course of my PhD project. This fun partially originated from the interesting subject that I was researching, but mostly came from the nice people around me that supported me throughout this adventure. Allow me to thank a couple of those people.

First of all, I would like to thank my supervisory team (which is how the university likes to call them). **Gerard**, you were always there when necessary and you have supported me at all times throughout the course of my project. Discussions and meetings were consistently fruitful and I have never met you differently than cheerful and positive, which was very motivating for me. I wish you all the best now that you are retiring from academia. The knowledge I gained about wind turbines mostly originates from **Wim**, so many thanks for patiently explaining me all the details of this fascinating machine. The intuition you have for the physics of a wind turbine is fascinating. During our countless meetings in Delft it was intriguing to combine this intuition with my mathematical thoughts to tackle some of the challenges wind energy is facing.

One of the nice persons that definitely shaped my PhD was the supervisor I met on an almost daily basis at our beloved coffee machine in the library: **Benjamin**. The suggestions you had on my work were always constructive and have significantly improved the quality of my results, so many thanks for this. Besides the mathematics, the sheer amount of fun we had during conferences, such as those in California, Munich, and Crete, is huge and made visiting conferences always something to look forward to. I will never forget your love for Runge–Kutta methods, traveling to countries far away, and cycling every now and then, so keep enjoying life as much as you do right now, especially since you have recently become a proud father.

During the course of my PhD and well before the start of my project I have been sharing my office in Amsterdam with **Prashant**. He was there at the start of my project and also right now at the end as paranymph, which is quite remarkable. Our shared love for good music (especially Vivaldi) to distract ourselves made my days at work really memorable. Maybe I should not only thank you, but also apologize right now for the

infinite number of bad puns I have fired at you over the last four years. Fortunately you could take your revenge by beating me with ping pong once in a while (but let me emphasize that I have also won a couple of times).

I would also like to thank **Barry** and **Richard** for the constructive discussions about my work and insightful experiences at conferences. A small part of the paper we wrote together has found its way into this thesis and sparked the idea that led to the most important results of this thesis. Related to this, I also acknowledge the support of **Lindert**, **Menno**, and **Armando** from DNV GL for patiently explaining load calculations to me and inviting me to Groningen and Hamburg to give a talk about my work. I still recall the beautiful hotel and venue we had in Hamburg for the BLADED workshop, which led to many interesting discussions with people from industry about the relevance of my work.

During my PhD project I have supervised **Hemaditya**, who successfully became a Master of Science at the end of his project at the CWI. The algorithms we developed were very interesting and led to many useful insights. Thank you for extending my work with new ideas and especially for finding some non-trivial mistakes in my articles! All corrections have found their way in my thesis, so indirectly you have contributed to the quality of this end result.

It always confuses people when I have to explain them I work in Amsterdam at a research institute and obtain my PhD from the university of Delft. Unfortunately this also complicates communication with both parties. However, thanks to the organizational help of **Nada** and **Sylvia**, everything went smooth throughout these four years. Counting on you saved me from a lot of stress, so many thanks for arranging all practicalities and forms that were necessary to finish this thesis.

Related to this, I was affiliated with two research groups and have therefore been traveling a lot from Eindhoven to the Wind Energy group in Delft and to the Scientific Computing group in Amsterdam. I always felt welcomed wherever and whenever I came, so many thanks to all group members in both Delft and Amsterdam. In particular, I would like to thank the people situated in the nice corner of the first floor of the CWI, which are **Anne**, **Daan**, **Jurriaan**, **Wouter**, and **Yous**, for our regular and irregular coffee and tea breaks with discussions about virtually everything we could think about. Maybe I should also acknowledge the Dutch railway company (de Nederlandse Spoorwegen) here, since they have also provided me an office over the course of these four years (for two hours per day to be precise, sometimes unfortunately a bit longer).

Ten slotte zou het onwijs lastig zijn geweest om dit project succesvol af te ronden zonder de onuitputtelijke steun van mijn familie. Mijn ouders, **Fred** en **Corry**, broer en zus, **Vincent** en **Esmée**, hun partners, **Angelique** en **Bryan**, en familie van mijn vriendin, **Gerrie**, **Caroline** en **Lisette**, waren altijd geïnteresseerd in mijn werk en ondersteunden mij waar ik ook naartoe ging. Mijn PhD was echter geheel onmogelijk geworden als **Melissa** er niet voor me was geweest de afgelopen jaren. Zonder je ondersteuning bij mijn vele en late treinreizen, je luisterend oor bij problemen of succesjes en je oneindige interesse in mijn werk was het me nooit gelukt zo ver te komen als dat ik nu ben. Als er iemand coauteur van dit boekje zou moeten zijn, dan zou jij de eerste zijn.