# ADJOINT BASED DESIGN OPTIMIZATION OF SYSTEMS WITH TIME DEPENDENT PHYSICS AND PROBABILISTICALLY MODELED UNCERTAINTIES

A Dissertation
Presented to The Academic Faculty

By

Komahan Boopathy

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2020

# ADJOINT BASED DESIGN OPTIMIZATION OF SYSTEMS WITH TIME DEPENDENT PHYSICS AND PROBABILISTICALLY MODELED UNCERTAINTIES

Approved by:

Dr. Graeme J. Kennedy
Daniel Guggenheim School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Brian J. German
Daniel Guggenheim School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Marilyn J. Smith
Daniel Guggenheim School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Dewey H. Hodges
Daniel Guggenheim School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Boris Diskin
Senior Research Fellow
*National Institute of Aerospace*

Date Approved: July 22, 2020

*எப்பொருள் யார்யார்வாய்க் கேட்பினும் அப்பொருள்
மெய்ப்பொருள் காண்ப தறிவு.*

திருக்குறள் 423

Though things diverse from divers sages' lips we learn,
'Tis wisdom's part in each the true thing to discern.

Thirukkural 423 (translation by G. U. Pope)

Dedicating this journey across probabilistic-space-time to all my teachers, who have directly and indirectly shaped my thinking and reside within; particularly the first and foremost teachers – Mom and Dad!

# ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Graeme Kennedy for his support, inspiration and encouragement throughout my graduate studies at the Georgia Institute of Technology. I am grateful to each one of the committee members for their guidance over the years.

I thank my parents, younger brother and sister who have always been supportive of my endeavors. Many thanks to my friends and fellow graduate students: Ting Wei Chin, Mark Leader, Jan Kiviaho, Kevin Jacobson, Siddarth Niranjan Babu, Nicholson Koukpaizan, Chunrong Ngoh, Aaron Fu, Rick Dong, Jordan Trout, Ruthvik Chandrasekaran, Dushhyanth Rajaram, Darshan Sarojini, Hossein Salahshoor, Joachim Hodara and Elena Garombo, for making my time at graduate school enjoyable.

I am thankful to Prof. Pavel Grinfeld for his excellent lectures on linear algebra which helped me build concrete intuitions around a lot of mathematical concepts that were formerly abstract to me. I am in love with the concept of *inner products* ever since listening to his lectures on the YouTube platform. I am also thankful to the late Professor Hans Petter Langtangen of the University of Oslo for strengthening my intuitions about approaching the subject of uncertainty quantification from a linear algebra perspective.

I would like to acknowledge the funding provided by the NASA Langley Research Center and the National Institute of Aerospace. I also thank the School of Aerospace Engineering for supporting me with Graduate Teaching Assistantships.

## TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# SUMMARY

For aerospace structures, failure can occur not only because of static adversities like divergence, but also due to time dependent issues like flutter and large vibrations. Therefore, the consideration of time-domain physics becomes essential during design. The physics-based design of aerospace systems involves solving partial differential equations to obtain metrics of interest that guide the design process. These differential equations contain unknown parameters that are sometimes difficult to be characterized as a deterministic value. The uncertainties in input parameters have a direct impact on the output metrics of interest which guide the system design process. To this end, optimization under uncertainty has evolved as a field that accounts for the effect of uncertainties, by propagating the effect of uncertainties through physics simulations.

For numerical optimization, the algorithms that do not use gradient information become computationally intractable as the number of design variables increases. Moreover, the numerical approximations of the gradients through the finite-difference or the complex-step methods are inefficient, for their lack of scalability with respect to the number of design variables. Therefore, efficient gradient evaluation techniques such as the adjoint method are needed for solving large scale optimization problems with practical turnaround times. However, because of the inclusion of time dependent physics, the corresponding time dependent adjoint equations needs to be formulated and implemented. Furthermore, the uncertainties need to be propagated through the time dependent physics and the adjoint sensitivity analysis framework. Due to the inherent complexities in the development of time domain physics and adjoint sensitivities analysis capabilities, the sampling-based methods are widely used for the propagation of uncertainties while the projection-based methods are less used.

This work presents enhanced implicit time marching methods for flexible multibody dynamics, to analyze the time dependent behavior of aerospace structures, and formulates the corresponding time dependent adjoint sensitivity analysis equations, to efficiently optimize designs using gradient based methods. The adjoint-based design capabilities are demonstrated with the structural optimization of a rotorcraft hub system. A newly developed semi-intrusive approach for projection is shown to fully reuse the underlying time-domain analysis and adjoint sensitivity analysis capabilities, for the projection-based propagation of uncertainties. Using this method, the stochastic residuals and Jacobians are formed implicitly from the deterministic counterparts that have been implemented apriori. The application of the semi-intrusive projection method is shown using a flexible robotic manipulator system modeled after the Canadarm. In the presence of uncertainties in the payloads, the Canadarm system experiences stresses that have a large variability. This work demonstrates the use of uncertainty quantification as a valuable tool for assessing the risk associated with such operating conditions.

# Part I

# Introduction

# CHAPTER 1
# EFFICIENT OPTIMIZATION UNDER UNCERTAINTY OF SYSTEMS WITH TEMPORAL PHYSICS

> *A product should be designed in such a way that makes its performance insensitive to variation in variables beyond the control of the designer.*
>
> *Genichi Taguchi*

We begin this Chapter with Genichi Taguchi's quote on producing robust and reliable designs. The motivating factors of the thesis are summarized and classified under three subject areas: uncertainty quantification, time dependent physics and adjoint sensitivity analysis. Finally, we outline the contributions of this thesis and present its organization.

## 1.1 Motivations

**(a) Uncertainty quantification:** The Federal Aviation Authority (FAA) airworthiness certification requires a factor of safety of 1.5 for aircraft structures with human occupancy [2]. The inclusion of a factor of safety as a certification requirement is a tacit acknowledgment of the ubiquitous presence of uncertainties that are beyond the scope of classical system design process. For aerospace systems, higher factor of safety implies heavier designs with increased operation costs for the entire life cycle of the system. Despite factor of safety stipulations in the design process, systems do fail (a risk concern) or perform in a degraded manner (a robustness concern), partly due to a lack of uncertainty assessments before designing the system. To this end, the fields of uncertainty quantification (UQ) and optimization under uncertainty (OUU) have evolved to rigorously address the effect of uncertainties in the design process. UQ addresses the mathematical representation and propagation of input uncertainties, whereas OUU addresses the mathematical aspects of formulating design/regulatory requirements as objective or constraint functions.

**(b) Temporal physics:** The mathematical models of physics can also be a contributing factor for unforeseen system behavior. For example, when fixed- and rotary-wing aeromechanical structures are designed without time dependent analysis of response (by using a static evaluation), the onset of many time dependent adverse effects such as limit cycle oscillations, buffeting, flutter, stall-induced vibration and rotor-shaft whirl can go unpredicted. Arguably, inclusion of time domain within system analysis is as important as uncertainty quantification; thus, time dependent mathematical models of physics along with mathematically modeled uncertain inputs encompass a superior representation of system behavior. The systems designed using such inclusive analyses will emerge better in terms of robustness and reliability.

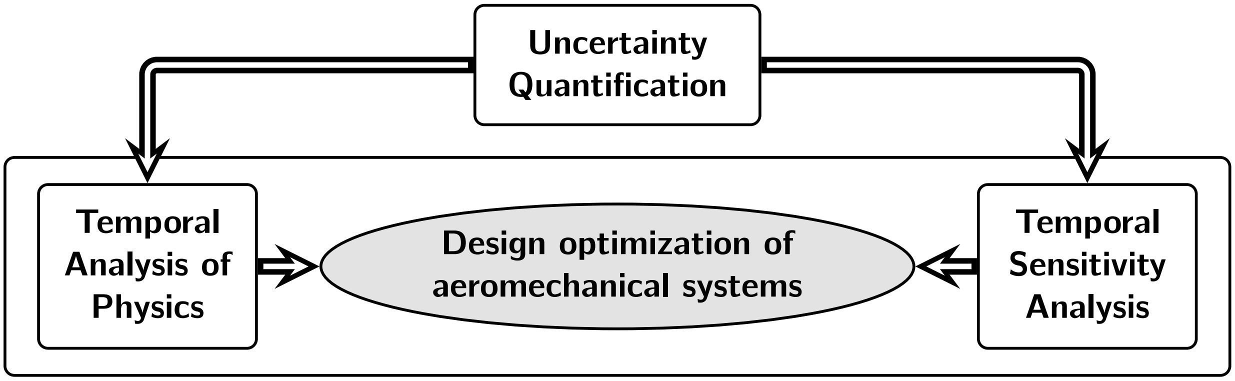

Uncertainty Quantification
Temporal Analysis of Physics
Design optimization of aeromechanical systems
Temporal Sensitivity Analysis

*Figure 1.1: An integrated design framework with temporal physics, uncertainty quantification and sensitivity analysis.*

**(c) Adjoint-based gradient evaluation:** Numerical optimization of large aeromechanical systems require gradient-based optimization techniques that are computationally efficient compared to techniques that do not use higher-order information. Therefore, an efficient evaluation of gradients is also an important ingredient to the UQ–OUU design process. The time dependent nature of physical analysis necessitates the development of time dependent sensitivity analysis equations. Altogether, a need for incorporation of temporal analysis of physics, uncertainty analysis and sensitivity analysis into a common design framework emerges naturally (see Figure 1.1).

## 1.2 Thesis Contributions

The main contributions of this thesis are as follows:

1. We present simpler time dependent analysis methods for flexible multibody systems, that can be used to assess the onset of time dependent issues like flutter or large vibrations while designing aeromechanical systems.

2. We contribute adjoint based gradient evaluation capabilities to address the issue of the scalability of optimization problems with respect to the design variables, in the context of time dependent simulations.

3. In the context of uncertainty propagation, we address the issue of "intrusiveness" of the stochastic Galerkin method by presenting a simpler technique to achieve projection in probabilistic space.

4. We present a stochastic Galerkin based OUU framework that can be used to solve probabilistic optimization problems, and provide information in the form of probabilistic moments that can be used for certification and quality assurance purposes.

The technical contributions align with the subject areas shown in Figure 1.1, and are summarized as *deterministic optimization* and *optimization under uncertainty* capabilities, in the reminder of this section.

### 1.2.1 Deterministic Finite Element Framework with Adjoint Sensitivities

The first contribution of this thesis is the development of high-fidelity simulation techniques and the implementation of adjoint-based derivative evaluation method for time-accurate flexible multibody dynamic simulations. These capabilities are implemented within the Toolkit for Analysis of Composite Structures (TACS), a parallel framework for finite element analysis [3], that is available as open source software[1]. The finite element formulations based on Lagrange's equation of motion were implemented, along with implicit time marching methods and adjoint formulations.

### 1.2.2 Stochastic Finite Element Framework with Adjoint Sensitivities

The second contribution of this work is the modular extension of the deterministic design optimization capabilities to include uncertainties through projection-based stochastic Galerkin technique. These capabilities are implemented as modular extensions to the TACS framework, and are available as open source packages PSPACE[2] and STACS[3].

**New Mathematical Techniques:** During the development of these frameworks for deterministic optimization and optimization under uncertainty of flexible multibody systems, we developed two mathematical techniques as listed in Table 1.1.

*Table 1.1: The developed mathematical techniques and their primary benefit.*

| | **Technique** | **Primary Benefit** |
|---|---|---|
| 1 | Generalized Newton–Raphson method for second-order nonlinear equations | facilitates the *direct* solution of the governing equations of flexible multibody dynamics in natural second-order form |
| 2 | Semi-intrusive stochastic Galerkin method | facilitates the reuse of time-domain physics and adjoint sensitivity analysis capabilities for creating a stochastic Galerkin framework for uncertainty propagation |

**Other Contributions:** The other notable contributions are listed as follows:

- We develop discrete adjoint sensitivity formulations for implicit multistep and multistage time marching methods based on abstract governing equations and functions of

[1] https://github.com/gjkennedy/tacs

[2] https://github.com/komahanb/pspace

[3] https://github.com/komahanb/stacs

interest. These equations are applicable to general second-order systems.

- We address the software architecture aspects alongside the mathematical developments, which is key for modular implementations.
- We demonstrate the adjoint-based design capabilities with the structural optimization of a rotorcraft system.
- We present the application of the semi-intrusive projection method using a flexible robotic manipulator system modeled after the Canadarm.

## 1.3 Thesis Organization

The remainder of this thesis is organized into parts and chapters as outlined in Table 1.2.

*Table 1.2: Organization of the thesis.*

| **Part** | **Chapter** | **Contents** |
|---|---|---|
| I | 2 | introduces and reviews the time marching methods for flexible multibody dynamics, the uncertainty quantification techniques and the sensitivity analysis methods that are needed for design optimization under uncertainty |
| II | 3 | provides the mathematical details of the adjoint and the direct sensitivity analysis methods on static (time independent) problems |
|  | 4 | provides the mathematical details of time dependent physical analysis and adjoint sensitivity analysis, in the context of multistep and multistage time marching methods |
|  | 5 | presents the deterministic optimization applications in the context of flexible multibody dynamics |
| III | 6 | presents the mathematical preliminaries necessary for the presentation of uncertainty propagation methods as inner products and corresponding quadrature approximations |
|  | 7 | presents the mathematical details of sampling and semi-intrusive projection approaches for uncertainty propagation, along with the software architecture for programming implementations |
|  | 8 | illustrates the semi-intrusive stochastic Galerkin method on simple time dependent systems and flexible multibody dynamics problems |
| IV | 9 | summarizes the results and contributions as well as outlines future research directions |

# CHAPTER 2

# AN OVERVIEW OF TEMPORAL, SENSITIVITY AND UNCERTAINTY ANALYSES

In this chapter, we review implicit time marching methods for flexible multibody dynamics as well as methods for sensitivity analysis and uncertainty quantification. Finally, the specific objectives of the thesis are discussed.

## 2.1 Temporal Analysis of Physics: Flexible Multibody Dynamics

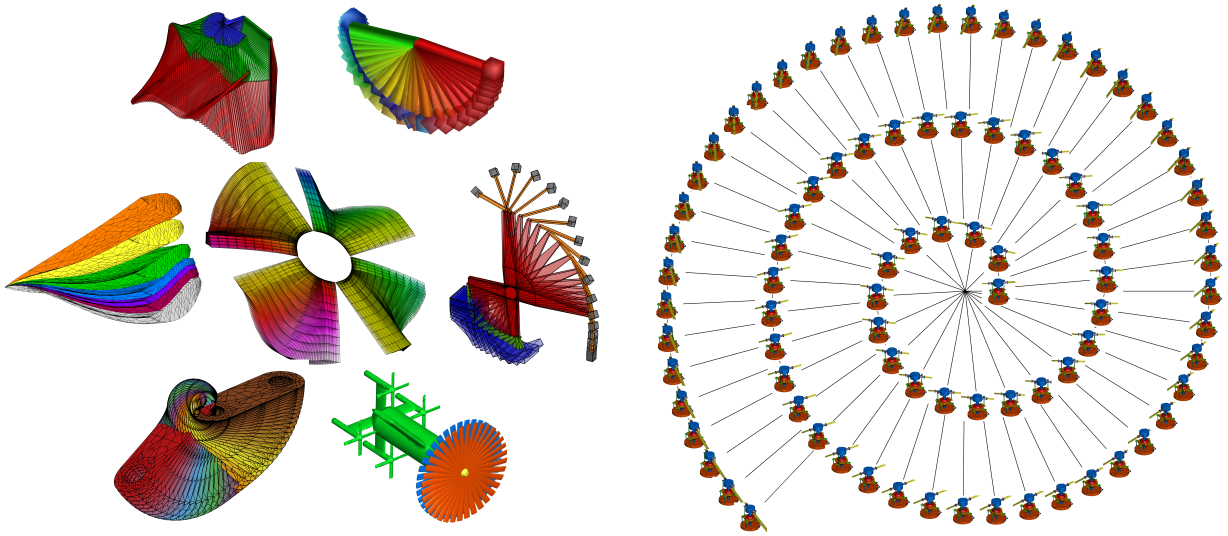


*Figure 2.1: Timelapse and timespirals depicting temporal evolution of dynamical systems.*

### 2.1.1 Abstract Form of Governing Equations

The temporal evolution of some flexible multibody systems are shown in Figure 2.1. Such systems can be studied by solving a system of nonlinear ordinary differential/algebraic equations of the form:

$$R(t, \xi, u(t, \xi), \dot{u}(t, \xi), \ddot{u}(t, \xi)) = 0, \tag{2.1}$$

where $u(t, \xi)$ is a function that describes the physical state of the system, along with functions describing the time rate of change: $\dot{u}(t, \xi)$ and $\ddot{u}(t, \xi)$. Here $t$ is the temporal variable and $\xi$ is the design variable. In principle, the abstract descriptor form (2.1) form facilitates the treatment of different formulations and discretizations of the governing equations, as well as governing equations for different physics under a common mathematical framework. For example, Equation (2.1) can be viewed as an abstract representation of time dependent processes resulting from:

- purely algebraic equations (e.g. spring mass damper system, Van der Pol oscillator) or
- algebraic equations resulting from spatial discretization (e.g. beam deformation model, Laplace equation) or
- algebraic equations of a particular physics resulting from different formulations (e.g. Newton–Euler method, Maggi's method, Euler–Lagrange method, Hamilton's principle [4, 5]).

Only the characterization (size and physical interpretation) of $u(t)$ differs from case to case, whereas the process of solving for $u(t)$ remains more or less the same; namely, linearization followed by iterations followed by time-stepping.

**Stiffness and Drifting:** In the context of dynamics, in presence of constraint equations, the system (2.1) represents a set of differential algebraic equations (DAEs), whereas in the absence of constraints it reduces to ordinary differential equations (ODEs). The ODEs and DAEs are collectively referred to as initial value problems (IVPs). There are some characteristic difficulties associated with solving DAEs when compared to ODEs. The presence of kinematic constraints make DAEs of flexible multibody systems numerically stiff to solve. The highest time derivative of the kinematic part of the equations is usually less than two, but the kinetic (dynamic) part of the equations contain second time derivatives, which leads to equations that contain varying scales of time. Therefore, the solution of DAEs is not as straightforward as the solution of ODEs, from a numerical solution perspective, and often requires specialized scaling of terms. Sometimes, the kinematic equations are differentiated to match the second derivative (see Haug et al. [6]) to transform DAEs to ODEs, but the satisfaction of true non-differentiated form of kinematic constraints is not guaranteed due to a phenomenon referred to as drifting. Bauchau and Laulusa [7] presents a review of constraint violation stabilization techniques that have been developed in the literature. In this work, the techniques to address the issue of drifting are not investigated; however, we enforce the constraints in index-2 form to ensure that drifting does not occur.

**Steps in Numerical Solution of ODEs/DAEs:** The major steps involved in the classical numerical solution of DAEs are:

1. converting the DAEs to first-order form,
2. choosing an explicit or implicit solution method, and
3. choosing a multistep or multistage derivative approximation hypothesis.

These steps are detailed next.

### 2.1.2 Conversion to First-Order Form : State-Space Representation

As noted previously, the second-order differential equations in time that model the dynamics of flexible multibody systems are of the form (2.1). The first step in classical solution approach is to define an equivalent first-order representation for (2.1) of the form:

$$S\left(t, \xi, v(t, \xi), \dot{v}(t, \xi)\right) = 0 \tag{2.2}$$

where $v(t, \xi)$ and $\dot{v}(t, \xi)$ are newly defined unknown functions. Effectively, the higher-order differential equations are transformed to equivalent first-order equations using algebraic transformations of the original unknown state functions $u(t, \xi), \dot{u}(t, \xi)$ and $\ddot{u}(t, \xi)$. This results in defining pseudophysical state functions $v(t,\xi)$ and $\dot{v}(t,\xi)$ whose codomain is larger than the codomain of $u(t,\xi)$; notably, the size of $v(t,\xi)$ is greater than the size of $u(t,\xi)$. Since numerical techniques for the solution of first-order initial value problems (IVPs) are well established and is implemented as a part of many numerical solution packages (for example, `EPISODE` [8], `ODEPACK/LSODE` [9] and `DASSL` [10]), the conversion to first-order form is justified in a practical sense. We emphasize that, it is not a fundamental mathematical requirement to solve the ODE in first-order form, but rather a conventional approach to utilize existing numerical libraries, solution algorithms, and proofs pertaining to first-order systems. The first-order representations are not necessarily unique due to flexibility (availability of numerous options) in transformation of variables, and can be from algebraically simple to cumbersome depending on the actual explicit form of (2.1).

**A Philosophically Different Classical Technique:** At this juncture, it becomes important to examine another classical numerical solution technique specifically developed for structural dynamics known as the Newmark [11, 12] method. The Newmark method deviates from converting to first-order form and operates directly on the second-order form of equations. The seminal authors and others attribute its stability, order of accuracy and numerical dissipation as suitable aspects for numerical solution of structural dynamics equations. Later, Chung and Hilbert [13] generalized the Newmark method to a class of methods referred to as Generalized$-\alpha$ method, where the choice of parameter $\alpha$ produces different schemes such as

1. Original Newmark
2. Hilber-Hughes-Taylor (HHT) method
3. Chung–Hilbert method (CH)
4. Wood–Bossak–Zienkiewicz (WBZ) method

The Generalized-$\alpha$ method features unconditional stability for specific choices of parameter $\alpha$ and the general order of accuracy is two (except Fox and Goodwin [11] with third order accuracy). Note that unconditional stability and higher-order accuracy is also a well known feature of Backwards Difference Formulas (BDF) [14, 15] widely used in the area of computational fluid dynamics (CFD), where the governing Navier–Stokes equations contain first-derivative in time. Also Runge–Kutta (RK) based methods having comparable stability and accuracy properties have been reported by Jameson et al. [16]. In general, when authors intend to use BDF/RK method the equations are in first-order form (2.2) and when Newmark/Generalized-$\alpha$ method is used the equations are in second-order form. For instance,

- the `SU2` [17] framework implements RK method for fluids (first-order equations in time) and Newmark method for structural dynamics (second-order equations in time),
- the `Metafor` [18] framework for the simulation of solids subject to large deformations as well as the `Dymore` [19] framework for flexible multibody dynamics implement Generalized-$\alpha$ method for time marching.

Thus, in the context of solving second-order system of equations, the main advantage of the Newmark method is that it is directly applicable for second-order equations (2.1), whereas other methods are employed on the first-order form of equations (2.2).

**Solving in natural higher-order form:** In this work, backward difference formulas (BDF), Runge–Kutta (RK) methods, Adams–Bashforth–Moulton (ABM) methods will be derived for governing equations in second-order form, enabling their direct application to DAEs and providing a common framework for adjoint-based derivative evaluation. Within the existing literature, we find that Haug et al. [6] extends the RK method for second-order systems in descriptor form. Their effort is in harmony with the principle that is emphasized here. However, the foundational principle of solving the second-order system without converting to first-order form is not directly suggested as a guiding principle by the authors of original Newmark method, or Generalized-$\alpha$ method or Haug et al. [6] who extended the RK method, although they seem to have used this principle. This work intends to explicitly introduce this guiding principle for solving IVPs directly in higher-order form, which will help:

- enhancing the body of time marching methods available for numerical solution of IVPs of classical and chaotic dynamical systems [20]
- circumventing the need to convert differential equations to first-order form which requires additional mathematical work

Therefore, we propose the development of a homogeneous body of numerical methods for time marching of flexible multibody dynamics, operating based on abstracted governing equations in second-order (2.1) (see Figure 2.2 for an illustration this idea). When the steps in solution process are formulated based on a common mathematical abstraction, the software implementation of these techniques become a simple and efficient. The mathematical abstraction (2.1) parallels the role of "Interfaces and Abstract Classes" in contemporary software development terms. We also take this opportunity to highlight that the success of object oriented software development is inherently related to the mathematical derivations; the former is imperative for the latter. The importance of this step is often naive overlooked by physicists and engineers while deriving equations.

### 2.1.3 Multistep and Multistage Methods

Time marching methods advance the physical state of the system step-by-step. A *step* is defined as advancing state functions from $t_{k-1}$ to $t_k$, whereas a *stage* can be viewed as an intermediate point in time domain between two steps, $\tau \in [t_{k-1}, t_k]$. DAEs contain time derivatives and therefore require a hypothesis for their numerical approximation. Multistep and multistage time-derivative approximation hypotheses emerge from a classification based on the time-level from which system state information is utilized (see Figure 2.3 for an illustra-

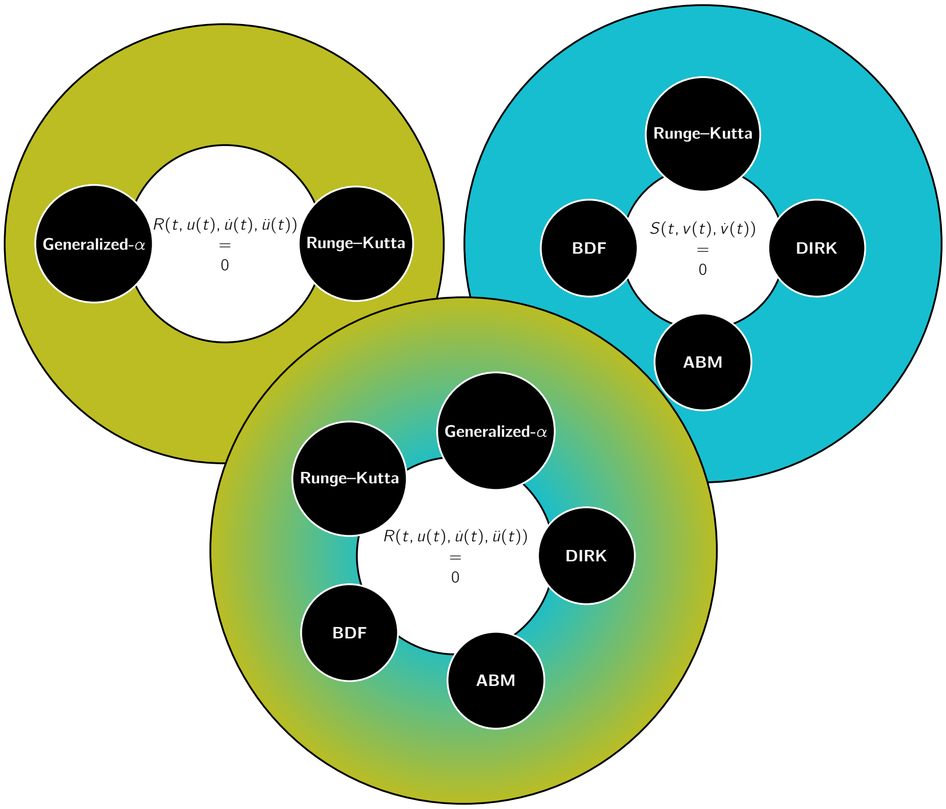


*Figure 2.2: Enhancement of body of numerical methods for the solution of flexible multibody dynamics equations in second-order form.*

tion). Methods such as Backwards Difference Formulas (BDF) [14, 15], Adams–Bashforth–Moulton (ABM) [21, 22] are regarded as multistep, whereas Runge–Kutta (RK) [23] and Diagonally Implicit Runge–Kutta (DIRK) [24, 25] are regarded as multistep methods. In general, multistage methods require more numerical work compared to multistep methods. The order of accuracy preservation becomes difficult for multistep methods with a lack of sufficient state history at the beginning of time marching process. Thus, multistep methods are non-self-starting whereas multistage methods are self-starting. A common start-up strategy is using multistage methods to generate system states for few initial time steps until enough states are available for multistep methods.

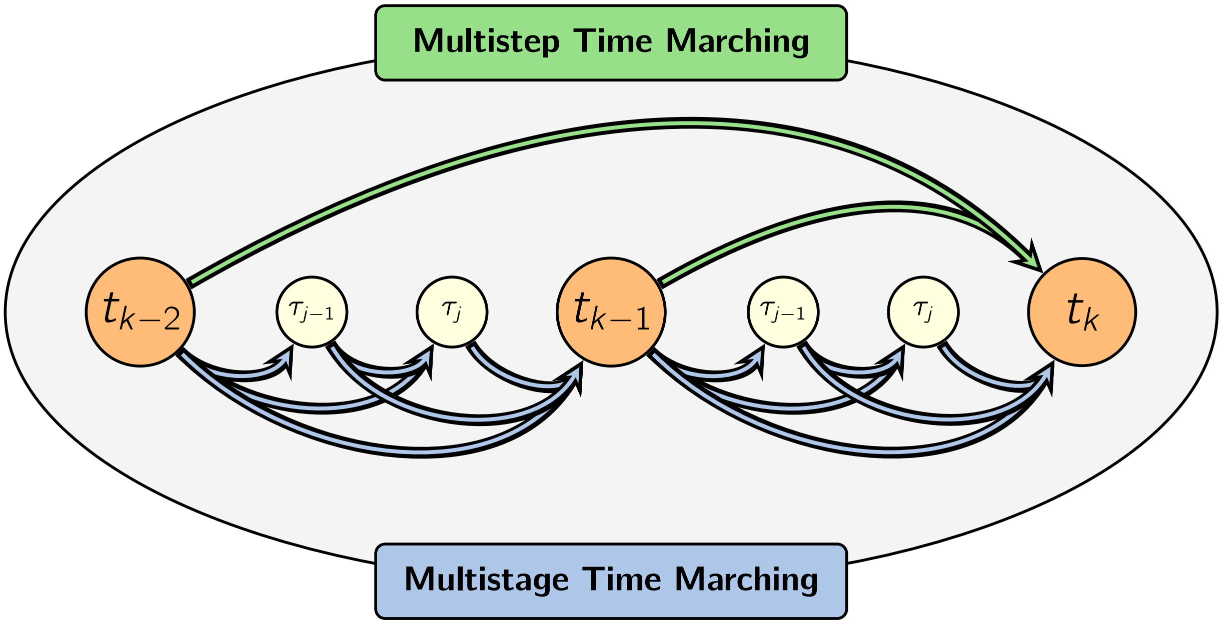


*Figure 2.3: Connections between steps and stages of multistep and multistage approximation methods of time derivatives.*

### 2.1.4 Explicit and Implicit Nonlinear Solution

The selection of a derivative approximation hypothesis allows casting the nonlinear system of differential-algebraic equations (DAEs) as nonlinear algebraic equations (time-derivatives are discretized). Now, the advancement of system state to next time level can follow explicit or implicit paths or some combination of both. Explicit time marching techniques advance the system state from one time level to another without solving system of nonlinear equations, whereas implicit methods have an intrinsic requirement of solving system of equations for time advancement. Although both methods come with comparable theoretical accuracies, the distinguishing factor is the superior stability of implicit schemes. In the context of flexible multibody dynamics, the stiffness of DAEs necessitate the use of extremely smaller time steps if an explicit method is used, in order to achieve stability in the solution process. However, larger time steps can be employed when implicit schemes are used, which turns out to be computationally efficient and robust in the context of solving DAEs. Gear [26] and Brenan et al. [10] discuss solution methods for stiff and non-stiff systems written in first-order form (2.2). A possible hybrid approach is to partition the DAEs into stiff and non-stiff parts, and to solve the stiff algebraic part using implicit integration schemes, and the non-stiff part using explicit methods. In this work, the focus is on implicit techniques for time advancement.

**Implicit Newton–Raphson Nonlinear Solution:** The Newton–Raphson iterative solution process is to linearize the nonlinear system and solve the resulting linear systems repeatedly. As long as computer implementations permit the evaluation of residuals and corresponding Jacobian matrices at each linearization point, the nonlinear system can be solved to determine the states for studying the temporal behavior of systems.

## 2.2 Techniques for Sensitivity Analysis

Let $f(\xi)$ be a function of interest (e.g. stress, failure) evaluated after the numerical solution of the physical state of aeromechanical systems $u, \dot{u}, \ddot{u}$, where $\xi$ is the design variable. Let $f(\xi)$ represent functions that are either integrated in time variable $t$ as

$$f(\xi) := \int_{t_{initial}}^{t_{final}} F\left(t, \xi, u(t,\xi), \dot{u}(t,\xi), \ddot{u}(t,\xi)\right)\ dt, \tag{2.3}$$

or functions evaluated at specific instance of time $t_k$ as

$$f(\xi) := F\left(t_k, \xi, u(t_k,\xi), \dot{u}(t_k,\xi), \ddot{u}(t_k,\xi)\right). \tag{2.4}$$

Some common techniques used to compute the derivatives of these functions of interest with respect to variables subject to design $\xi$ are reviewed in this section. Figure 2.4 presents a characteristic classification of derivative evaluation methods.

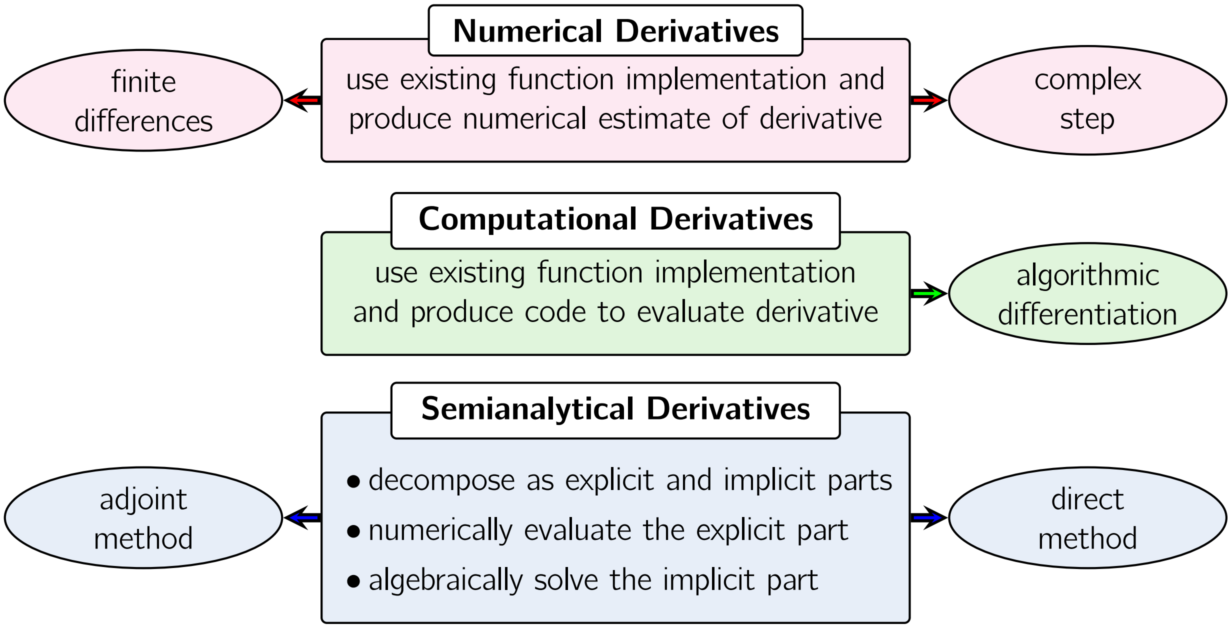

Numerical Derivatives
use existing function implementation and produce numerical estimate of derivative
finite differences
complex step
Computational Derivatives
use existing function implementation and produce code to evaluate derivative
algorithmic differentiation
Semianalytical Derivatives
• decompose as explicit and implicit parts
• numerically evaluate the explicit part
• algebraically solve the implicit part
adjoint method
direct method

*Figure 2.4: Classification of derivative evaluation methods based on principles followed.*

### 2.2.1 Numerical Methods

The numerical methods for sensitivity analysis work without the need for an explicit mathematical expression for derivative. The only requirement is being able to evaluate the function of interest $f(\xi)$ for input $\xi$.

#### *2.2.1.1 Finite Difference Method*

The finite difference method is a simple numerical method to approximate derivatives. Using this method, the first derivative of function of interest is approximated to first and second-order accuracy respectively as

$$\begin{aligned} \frac{df(\xi)}{d\xi} &= \frac{f(\xi+\Delta\xi)-f(\xi)}{\Delta\xi} + \mathcal{O}(\Delta\xi) \\ \frac{df(\xi)}{d\xi} &= \frac{f(\xi+\Delta\xi)-f(\xi-\Delta\xi)}{2\Delta\xi} + \mathcal{O}(\Delta\xi^2). \end{aligned} \tag{2.5}$$

Similarly higher-order approximations of the first derivative can be obtained using generalized forward, backward or central difference stencils. In many aeromechanical systems, there are several thousand design variables; thus finite difference method is limited to smaller optimization problems.

**Obtaining higher derivatives:** The concepts of finite difference method are general and applicable to higher derivatives of function with respect to $\xi$ as well. For example, the second derivative of function is approximated using central differences as

$$\frac{d^2 f(\xi)}{d\xi^2} = \frac{f(\xi + \Delta\xi) - 2f(\xi) + f(\xi - \Delta\xi)}{\Delta\xi^2} + \mathcal{O}(\Delta\xi^2). \tag{2.6}$$

The accuracy of finite difference method is strongly influenced by the choice of the step size $\Delta\xi$ and numerical loss of precision due to subtractive cancellations. Its computational cost scales linearly with the number of design variables, making this method computationally unsuitable for functions with large number of input variables.

### *2.2.1.2 Complex Step Method*

The complex step approximation [27, 28] of first derivative is obtained by perturbing the imaginary part of function input as

$$\frac{df(\xi)}{d\xi} = \frac{f(\xi + \Delta\xi i)}{\Delta\xi} + \mathcal{O}(\Delta\xi^2), \tag{2.7}$$

where the design variable $\xi + 0i$ is perturbed by adding an imaginary component $0 + \Delta\xi i$. The complex-step method is second-order accurate; therefore the truncation error of associated Taylor series expansion decreases quadratically when the perturbation size is reduced. Unlike the finite-difference method, this method does not suffer from lack of precision due to subtractive cancellation (as there is no subtraction involved), which enables the use of very small perturbation step sizes to produce highly accurate derivative estimates. However, the complex-step method is computationally more expensive compared to the finite-difference method due to the use of complex number arithmetic. In Figure 2.5 the accuracy of derivative approximations obtained using the two numerical methods (finite differences and complex step) are compared on a test function. Note that the slope of lines correspond to the order of accuracy of the approximation. It can be seen that, for the finite difference methods, subtractive cancellations take effect as step sizes get smaller. Unlike the complex step method, there is always a practical limit to the accuracy of derivative approximations when finite differences are used.

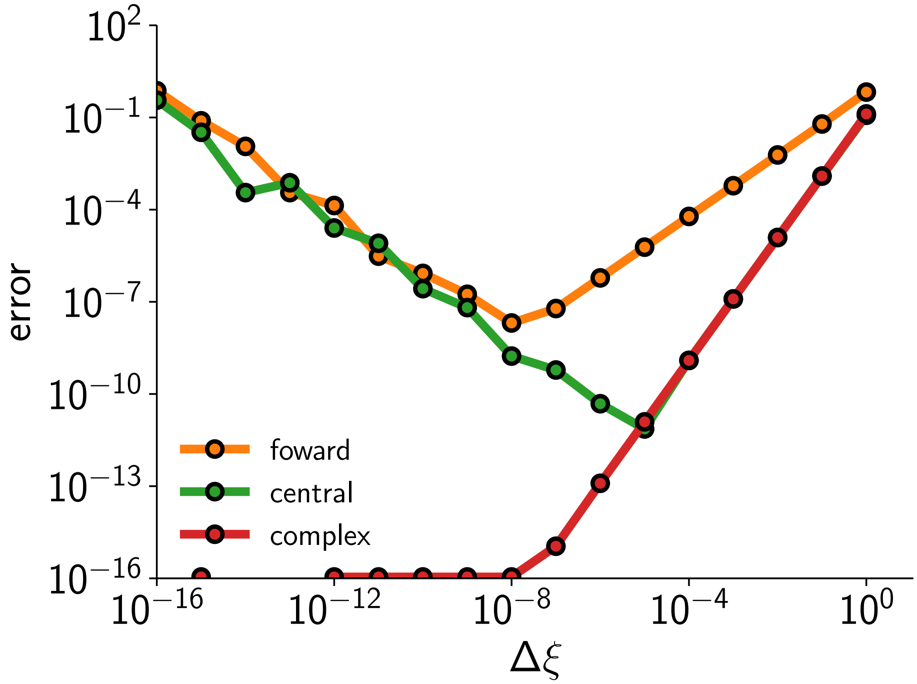


*Figure 2.5: Absolute error in approximated derivatives obtained from FDM and CSM for decreasing perturbation sizes.*

**Obtaining higher derivatives:** Higher dimensional numbers such as quaternions or hyperdual numbers can be used to approximate higher derivatives. However, this approach is not common in numerical and scientific computing libraries. Thus the idea of attributing additional imaginary dimensions to real numbers to compute higher derivatives is rather less explored, but there have been some aerospace applications of this technique (see Fike and Alonso [29], Fike et al. [30]).

### 2.2.2 Computational Methods

The computational methods act on the principle of obtaining the computer code for evaluating the derivatives from the computer code of the function itself.

#### *2.2.2.1 Algorithmic (Automatic) Differentiation*

Automatic differentiation (AD) is based on the application of the chain rule of differentiation to the computer code evaluating a function of interest. This approach produces computer code to evaluate first- and second-derivatives of the function. When the computer code is executed derivatives that are accurate to machine precision are obtained. AD methods are a promising avenue for research in obtaining sensitivities and there have been many applications of this method within and outside aerospace research [31–38]. The generated code to compute derivatives may or may not be in its algebraically simplified form and thus the code may not be optimal in terms of number of floating point operations (FLOPS).

### 2.2.3 Semianalytical Methods

The semianalytical methods decompose the total derivative as a combination of partial derivatives that are explicitly known (or approximated) and total derivatives that are implicitly solved using algebraic solution techniques. Mathematically, these methods can be derived as follows:

$$\begin{aligned}\frac{df(\xi)}{d\xi} &= \overbrace{\frac{\partial f(\xi)}{\partial \xi}}^{explicit} + \overbrace{\frac{\partial f}{\partial q}\frac{dq}{d\xi}}^{implicit}\\ &= \frac{\partial f(\xi)}{\partial \xi} - \frac{\partial f}{\partial q}\left[\frac{\partial q}{\partial R}\right]\frac{\partial R}{\partial \xi} \qquad (2.8)\\ &= \frac{\partial f(\xi)}{\partial \xi} - \frac{\partial f}{\partial q}\left[\frac{\partial R}{\partial q}\right]^{-1}\frac{\partial R}{\partial \xi}\end{aligned}$$

In practice, we can compute the residual $R$ and the Jacobian matrix $\left[\frac{\partial R}{\partial q}\right]$ and can use the numerical inverse of the Jacobian matrix $\left[\frac{\partial R}{\partial q}\right]^{-1}$ in place of $\left[\frac{\partial q}{\partial R}\right]$. The analytical methods are divided into two categories based on the setup of algebraic equations as direct and adjoint-variable methods as

$$\begin{aligned}\frac{df(\xi)}{d\xi} &= \frac{\partial f(\xi)}{\partial \xi} - \overbrace{\frac{\partial f}{\partial q}\left[\frac{\partial R}{\partial q}\right]^{-1}}^{adjoint\ \lambda}\frac{\partial R}{\partial \xi} \\ &= \frac{\partial f(\xi)}{\partial \xi} - \frac{\partial f}{\partial q}\underbrace{\left[\frac{\partial R}{\partial q}\right]^{-1}\frac{\partial R}{\partial \xi}}_{direct\ \phi}.\end{aligned} \tag{2.9}$$

The semianalytical methods provide us a systematic way to evaluate derivatives numerically. This process involves the solution of a linear system of equations to determine the implicit contributions. These analytical methods are based on the assumption that the partial derivatives are known whereas the total derivatives are not obtainable by analytical means. When even the partial derivatives are difficult to obtain or algebraically cumbersome, AD methods are used to supply them to the adjoint or direct sensitivities framework. The finite difference method can also be used for the purpose of providing partial derivatives at the expense of speed, scalability and accuracy.

### *2.2.3.1 Direct Sensitivity Method*

From Equation 2.9 we can see that the direct method defines decomposition coefficients as

$$\phi_i = \left[\frac{\partial R}{\partial q}\right]^{-1}\frac{\partial R}{\partial \xi_i}. \tag{2.10}$$

The direct method is computationally the most efficient method for large number of output functions, as the linear system (2.10) is independent of the number of functions $f(\xi)$. From a different point of view, it requires the solution of a linear system governing the direct sensitivity variables, for each component of the vector of design variables $\xi_i$. Therefore, the computational cost of the direct method grows proportional to the number of design variables. In many applications, not all design variables are independent of each other. For example, the design variables are constrained to manufacturing and aesthetic considerations such as smoothness and curvature. In such cases, there is a scope to reduce the number of effective design variables through formation of design variable groups, and liking mechanisms among groups. Thus, the conjunction of direct sensitivity method with design variable linking approaches [39] can make this method more practical. The applications of the direct method can be found in the works of Belegundu and Arora [40], Adelman and Haftka [41], Haftka and Adelman [42], Bhalerao et al. [43] and Dopico et al. [44].

### *2.2.3.2 Adjoint Variable Method*

From Equation 2.9 we can see that the adjoint method finds decomposition coefficients as

$$\lambda_j = \frac{\partial f_j}{\partial q} \left[ \frac{\partial R}{\partial q} \right]^{-1} . \tag{2.11}$$

The adjoint method is complementary to the direct method; it requires the solution of a linear system for each output function of interest $f_j(\xi)$. The computational cost of computing the derivative of the functions of interest using this method is nearly independent of the number of design variables. However, the computational cost grows proportional to the number of functions of interest (indexed as $j$). Consequently in cases where the number of functions is large, the adjoint method can become expensive. This is a limiting concern in structural design based on strength criteria where a large number of stress constraints may be required. In such cases, constraint aggregation methods [45, 46] can be used to reduce the number of function, thereby reducing the gradient evaluation cost. The adjoint method has been applied to structural [3, 40–42, 47], aerodynamic [48–51], coupled aeroelastic [52, 53] and flexible multibody dynamics cases [44, 54–56]. Cao et al. [54] presented general adjoint methods for differential algebraic equations in first-order systems, or systems that have been reduced to first-order form, with applications to multibody dynamics. Nachbagauer et al. [56] presented an adjoint method for multibody dynamics with focus on applications for inverse dynamics and parameter identification for rigid body problems. Dopico et al. [44] developed an approach for the sensitivity analysis of multibody systems based on Maggi's formulation of the governing equations using the direct and adjoint methods. Ding et al. [55] presented an adjoint method for computing the second derivative of functions of interest.

## 2.3 Design in the Presence of Uncertainties

Designing systems in the presence of uncertainties can be viewed as composed of two main phases: uncertainty quantification (the analysis phase) and optimization under uncertainty (the design phase). The uncertainty quantification (UQ) phase addresses the mathematical aspects of the uncertainty analysis, whereas the optimization under uncertainty (OUU) phase addresses the mathematical aspects of formulating design/regulatory requirements as objective or constraint functions. Figure 2.6 illustrates the UQ–OUU process with high level choices that one can make at different stages of the process. This section reviews the pertinent subject along the lines of this classification.

### 2.3.1 Uncertainty Quantification and its Stages

Uncertainty quantification is a process through which the effects of uncertainties on the performance of systems are analyzed. It is common to model aeromechanical and many engineering systems as partial differential equations (PDEs). For example, Euler–Lagrange equations are used to describe the dynamics and vibration of structures such as aircraft wings, rotor blades, bridges and buildings. Uncertainties are an inherent part of these mechanical systems as available input data to PDE models is incomplete and uncertain. Therefore the mathematical models of these systems should account for the presence of such uncertainties through uncertainty quantification (UQ) techniques [57–64]. The process of UQ can be broken down in to three stages as:

1. characterizing the source and form of uncertainties as mathematical functions (e.g. distribution types, intervals)

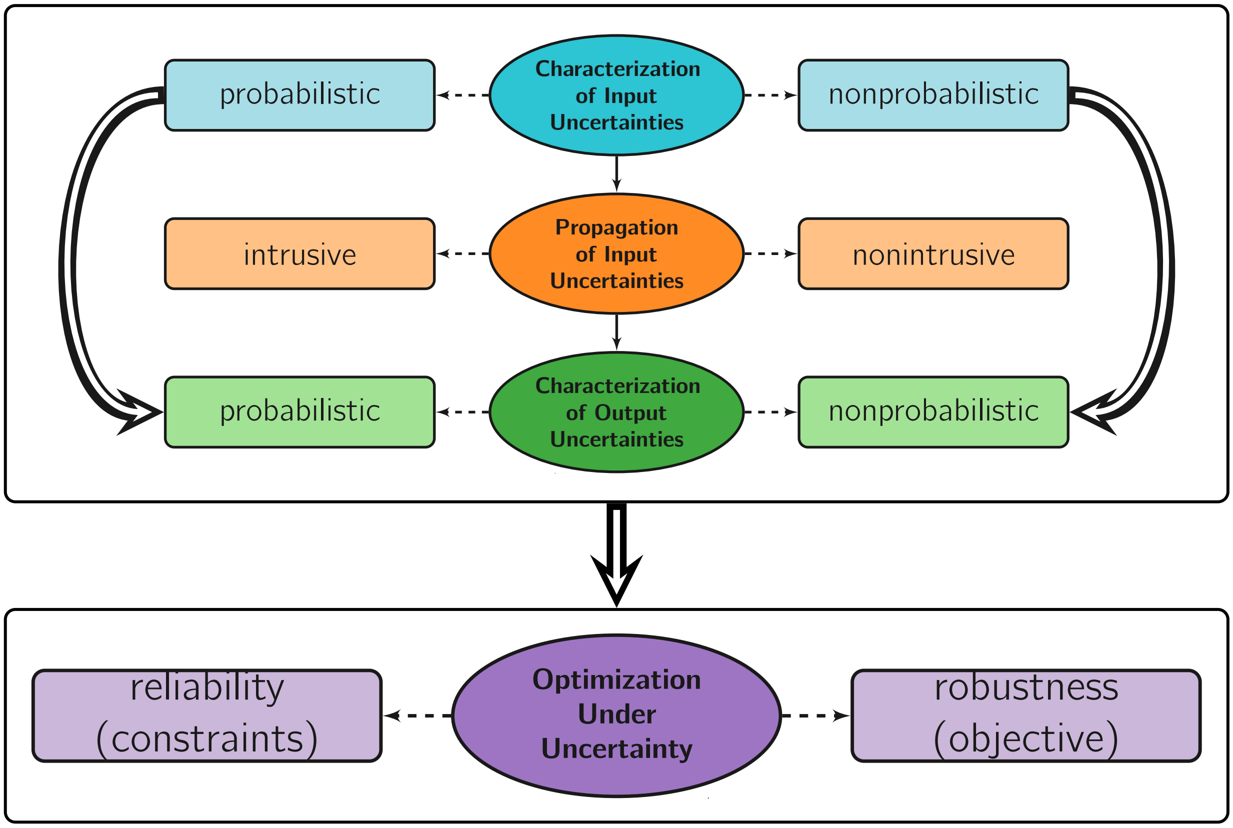

Characterization of Input Uncertainties
probabilistic
nonprobabilistic
Propagation of Input Uncertainties
intrusive
nonintrusive
Characterization of Output Uncertainties
probabilistic
nonprobabilistic
Optimization Under Uncertainty
reliability (constraints)
robustness (objective)

*Figure 2.6: A view of optimization under uncertainty process.*

2. propagating the input uncertainties through the mathematical model of aeromechanical systems

3. characterizing the behavior of output functions of interest (e.g. computing mean, variance, best and worst case behaviors, probabilities of failure)

#### *2.3.1.1 Stage I: Characterization of Input Uncertainties*

In the setting of partial differential equations, these uncertainties are a part of input functions, that collectively refer to the functions describing the distribution of coefficients and physical properties (e.g. material properties, viscosity), forcing functions (e.g. lift distribution on wing, controller input), and initial as well as boundary conditions. These uncertainties can be characterized probabilistically (uses probability theory) or nonprobabilistically (does not use probability theory).

**(1) Probabilistic Characterization (Aleatory Variables):** When deterministic specification of these input functions become difficult due to the lack of sufficient information, then probabilistic specification in the form of probability density functions (PDFs) can be beneficial. For example, instead of specifying a value for the representative force acting on a mechanical structure, the probability distribution function of force could be a more relevant model of the real scenario. When the input functions are probabilistically specified, the PDEs that operate on these input functions naturally inherit a probabilistic domain, $\mathcal{Y}$, along with the original spatial domain, $\mathcal{X}$, and temporal domain, $\mathcal{T}$. The variables from the probabilistic domain are referred to as random variables, analogous to spatial and temporal variables from respective domains. These random variables can arise naturally in the direct specification of PDE coefficients as random variables, or indirectly from the spatial and temporal discretization of correlated and uncorrelated random fields. Both sources are special cases of the general scenario where a vector of random variables are present in the problem (see Gunzburger [64]). Since the input functions contain an additional probabilistic domain, the deterministic PDEs that operate on these inputs, as well as the output metrics of interest, acquire the probabilistic domain and become stochastic partial differential equations (SPDEs). This naturally gives rise to the need for development of numerical methods for partial differential equations with random input functions. It is worth noting that SPDEs contain derivatives only in spatial and temporal variables; there are no derivatives in terms of random variables. Thus, from a vector-space point of view, we only need to find a set of basis functions to span the probabilistic space, where we can decompose probabilistic processes. This is identical in principle to finding finite-element basis functions to represent distribution of spatial processes, which analogously extends to finding orthonormal basis functions to represent probabilistic processes in this context of probabilistic uncertainty quantification. The mathematical details of this process is described in Chapter 6.

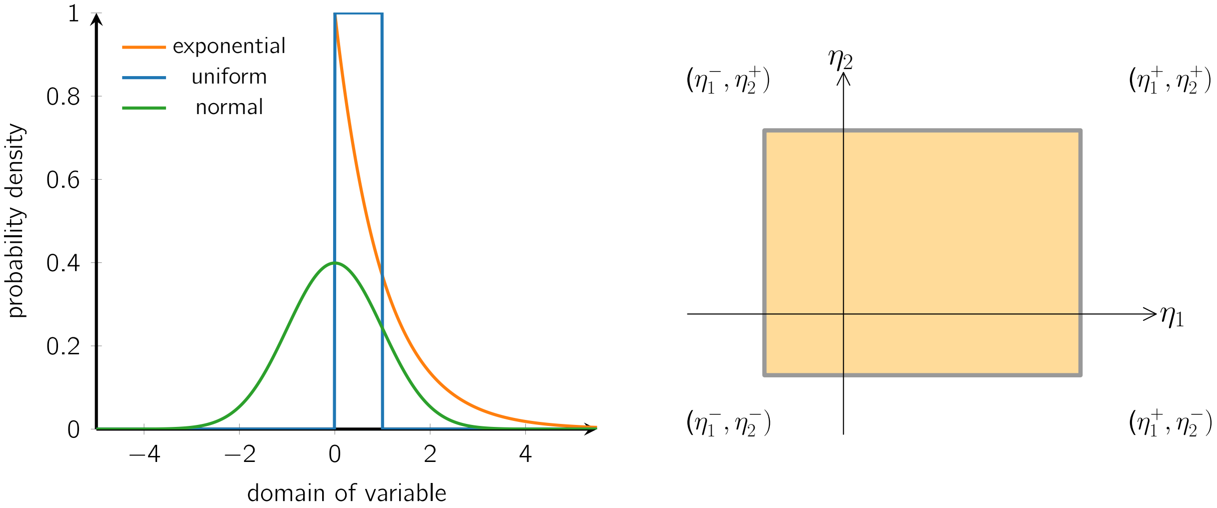


*Figure 2.7: Probabilistic and non probabilistic modeling of uncertainties.*

**(2) Nonprobabilistic Characterization (Epistemic Variables):** Sometimes, it is difficult to associate probability information with random variables due to lack of sufficient data. This happens because a large amount of empirical data is necessary to predict the distribution type in first place. For example, airline operators can predict the distribution of baggage weights, if they collect and store this data beforehand during check-in. Atmospheric data such as pressure, temperature, humidity *etc.* are usually stored in databases and are available for UQ applications. When data is not available, nonprobabilistic approaches such as possibility theory, interval analysis, convex modeling and evidence theory (see Keane and Nair [2]) are used. The simplest non-probabilistic approach is the interval representation of input uncertainties. Here, the random variable can take any value within the specified interval but the underlying probability distribution is unknown. See Figure 2.7 for an example of modeling uncertainties in inputs as probability distributions and intervals.

In summary, probabilistic approaches are apt for modeling aleatory uncertainties featuring an abundance of data and non-probabilistic approaches are suitable for epistemic uncertainties suffering a data scarcity.

#### *2.3.1.2 Stage II: Propagation of Input Uncertainties*

Uncertainty propagation is the second and predominant step in uncertainty analysis. It is performed using non-intrusive sampling-based and intrusive projection-based methods (see Figure 2.6). A high level review of these techniques are presented in the following paragraphs, whereas the mathematical details are deferred to Chapter 6.

**(1) Stochastic Sampling Methods:** The first class of techniques for uncertainty propagation are based on the idea of sampling. Sampling based techniques, collectively referred to as stochastic sampling methods (SSMs) rely on repeated solutions of the deterministic PDE at specified values of uncertain parameters from the probabilistic domain. Since this approach does not mandate any changes to the existing source code of the PDE solver, sampling-based techniques are referred to as non-intrusive [65–67]. The most-widely known sampling based technique for uncertainty propagation is the Monte Carlo (MC) sampling [68, 69]. The MC method draws samples at random and it is the only method that does not suffer from the curse of dimensionality (the convergence is independent of the number of random variables), but the rate of convergence is rather limited to $\mathcal{O}(1/\sqrt{M})$, where $M$ is the number of samples. A better selection of samples is offered by quasi-MC sampling methods (e.g. latin hypercube sampling), but at the cost of incurring a dependence on the number of variables and thus prone to the curse of dimensionality. The other type, namely, the quadrature sampling (also referred to as stochastic collocation) [70, 71], exploits the idea of Gaussian quadrature rules in the selection of sample points. This idea relies on the smoothness of interpolating polynomials and thus may not be suitable for functions with discontinuities. Even more restricting is the extension of one-dimensional quadrature rule to multiple dimensions using tensor product or similar rules, which leads to a very large number of points. In order to reduce the number of quadrature points, sparse quadrature methods have been proposed [64, 72, 73]. Since the reduction in number of quadrature points is achieved by exploiting the smoothness properties, these methods are not suitable for nonsmooth processes. Another approach is to build surrogate models [74–80] that are trained using a limited set of points (random, quasi-random or quadrature) and then replacing expensive deterministic solutions of PDE with inexpensive evaluations of the surrogate model. Sometimes the gradient information is also used in the construction of surrogate models alleviating the curse of dimensionality to an extent [76]. These SSMs are known to offer great flexibility in using deterministic codes as black-box solvers, and are thus widely used within the uncertainty analysis literature. Figure 2.8 illustrates the random, quasirandom and quadrature selection of samples from a two-dimensional random space.

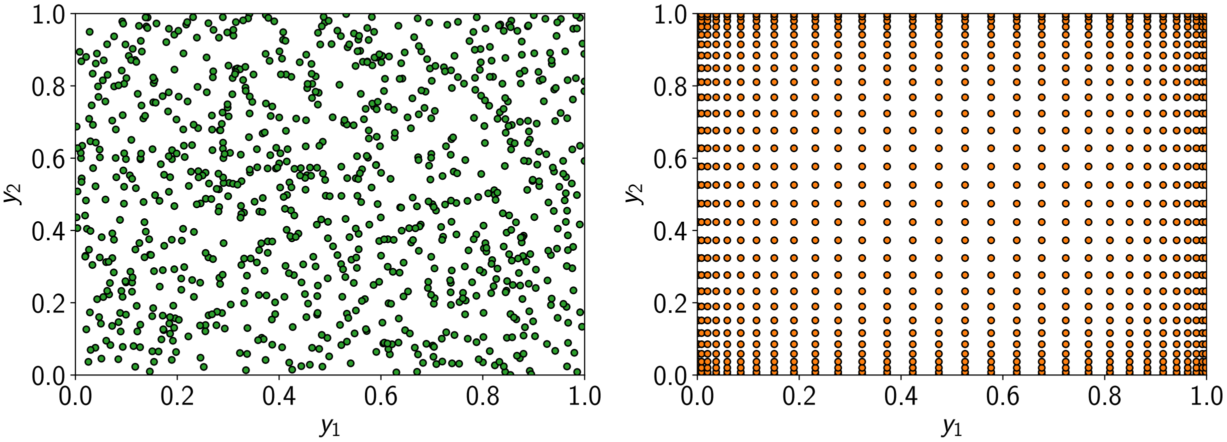


*Figure 2.8: Selection of samples using random and quadrature sampling methods to evaluate multidimensional integrals.*

**(2) Stochastic Galerkin Methods:** The second class of techniques for uncertainty propagation are based on the idea of Galerkin projection in probabilistic space and are collectively referred to as stochastic Galerkin methods (SGMs) [61, 81, 82]. Based on the construction of approximation to the probabilistic space, SGMs can be further classified into a few subcategories. The use of globally orthonormal polynomials for the approximation of probabilistic space has led to the development of methods based on spectral expansion [72, 83], where the entries in basis set have global support. This approach is also referred to as polynomial chaos expansions in the literature. Since the basis functions have global support the spectral expansion type methods are not the ideal choice if there exists discontinuities of the solution in terms of probabilistic parameter space. This motivates the use of basis functions with local support, similar to localized finite-element type constructions that can treat discontinuities. Based on the spatial discretization method some approaches are referred to as stochastic finite element method [57–59, 84–86] and stochastic finite volume methods [87]. All these SGMs operate on the principle of projecting (decomposing) stochastic functions in probabilistic (stochastic) space using *weighted inner products*, where the weighing functions are the probability density functions of the random variables. These inner products are defined as multidimensional integrals, and are numerically approximated using aforementioned multidimensional quadrature rules. Thus, multidimensional quadrature rules are used in both sampling-based and projection-based uncertainty propagation approaches. The SGMs differ from SSMs in that they directly solve the SPDEs instead of solving the deterministic PDE multiple times. The SGM is amenable for the development of specialized algorithms aiming to exploit the nature of equations in stochastic solvers that perform computations in an efficient manner. However, this development requires significant effort in terms of specialized solvers, thus leading to its classification as intrusive methods [72]. The semi-intrusive approach for stochastic projection presented in this work is aimed to alleviate this difficulty and simplify the implementation process.

In summary, the sampling-based methods are simple to use, as they treat the entire deterministic solution framework as black-box. However, projection-based methods require explicit source code modifications to perform integration in probabilistic spaces. Therefore, the application of projection-based methods is inhibited due to the extra effort involved in code development.

#### *2.3.1.3 Stage III : Characterization of Output Uncertainties*

The step of characterization of output uncertainties follow after the propagation of input uncertainties through the system models and the evaluation of metrics of interest. This stage is dependent on the first stage of uncertainty quantification; if nonprobabilistic methods are used to represent input uncertainties, then only nonprobabilistic information can be used to describe the behavior of outputs of the system. For example, when nonprobabilistic input bounds are processed into the analysis model, only bounds on the output metrics of interest can be constructed. Similarly, when inputs are probabilistically modeled, then probability distribution of the outputs can be obtained, along with probabilistic moments such as mean, variance and standard deviation. This output information can be used to formulate optimization under uncertainty problems. Figure 2.9 illustrates this coupled input–output characterization process.

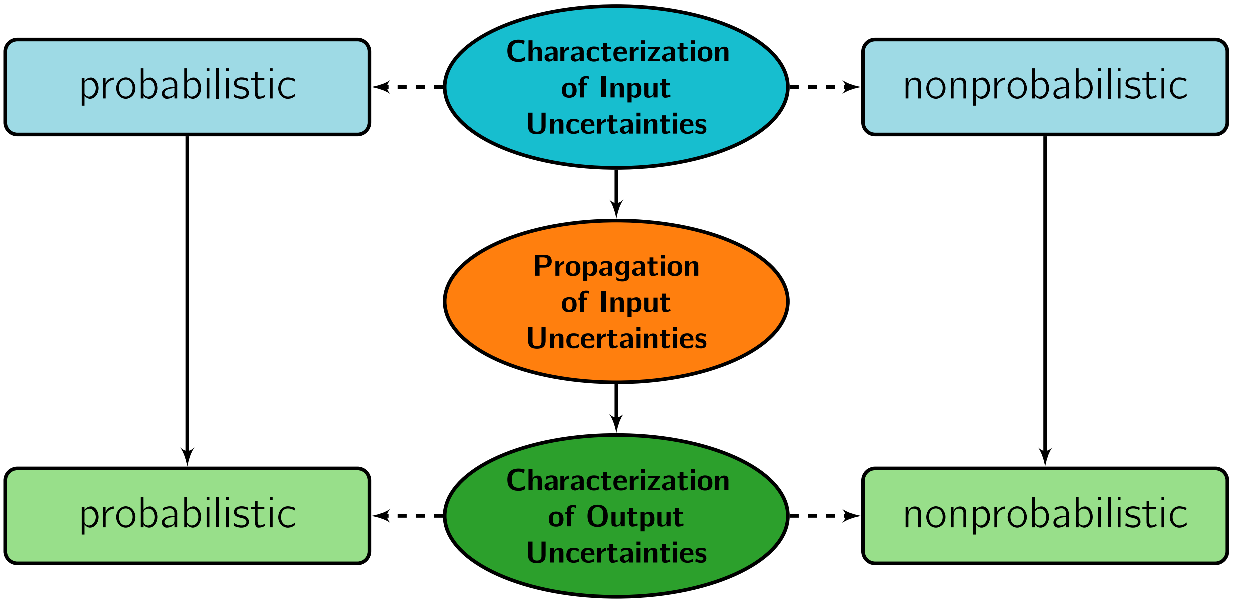

probabilistic
Characterization of Input Uncertainties
nonprobabilistic
Propagation of Input Uncertainties
probabilistic
Characterization of Output Uncertainties
nonprobabilistic

*Figure 2.9: Characterization of output uncertainties based on the characterization of input uncertainties.*

### 2.3.2 Optimization Under Uncertainty

The inclusion of uncertainty analysis within numerical optimization process is referred to as optimization under uncertainty (OUU). Within the OUU literature problems are typically classified as robustness-based formulations [88–105] and reliability-based formulations [106–108]. Table 2.1 provides a comparative summary of these formulations. Regardless of how the OUU problems are verbally named, the actual nature of the OUU problem (robust,

*Table 2.1: Summary of robust and reliability optimization.*

| Characteristic | Robust Optimization | Reliability Optimization |
|---|---|---|
| definition | a product design approach that facilitates reduction of performance variation | a product design approach aiming to reduce the probability of failure as much as possible |
| area of emphasis | the central part of probability distributions are studied (high probability events) | tail end of the probability distributions are investigated (low-probability events) |
| objective function | minimize the probability of failure | minimize the variance of objective function |
| computes | statistical moments and probability distributions | involves the computation of probabilities of rare events |

reliable, or both) is defined by the mathematical statement of the objective and constraint functions. In this section, first we introduce a general optimization problem without the inclusion of uncertainties and later compare it to the problem statement where uncertainties are included.

### 2.3.2.1 Deterministic Formulation

A deterministic optimization problem can be written as

$$\begin{aligned} \underset{\xi}{\text{minimize}} \quad & F(\xi) \\ \text{subject to} \quad & G(\xi) \leq 0 \\ & H(\xi) = 0 \\ \text{require} \quad & \frac{dF(\xi)}{d\xi}, \frac{dG(\xi)}{d\xi}, \frac{dH(\xi)}{d\xi} \end{aligned} \tag{2.12}$$

where $\xi$ are the uncertain design variables, $F(\xi)$ is the objective function, $G(\xi)$ are the inequality constraint functions and $H(\xi)$ are the equality constraint functions. The need for the derivative information arises with the choice of using gradient-based optimization algorithms, that are computationally efficient for large problems.

#### *2.3.2.2 Probabilistic Optimization Under Uncertainty Formulation*

Historically, OUU problems have evolved separately as robust or reliable formulations. However, when designing systems in the presence of uncertainties, the designer is concerned about both robustness and reliability aspects of the design. For instance, the airplane should be designed to ensure fuel efficiency amidst of uncertain wind gusts (robustness) without compromising on the safety aspects (reliability). We find that robustness discussions arise on the side of objective function whereas reliability discussions arise on the side of constraints. Figure 2.10 explains this idea along with mathematical statements that effect them. These mathematical statements are applicable only for probabilistically modeled uncertainties, whereas non-probabilistic models have different mathematical statements. The probabilistic moments such as the mean $\mathbb{E}[F(y(\xi))]$, variance $\mathbb{V}[F(y(\xi))]$, standard deviation $\mathbb{S}[F(y(\xi))]$ and probability $\mathbb{P}[G(y(\xi)) \leq 0]$ need to be evaluated to formulate a probabilistic OUU problem.

**Design Variables:** The design variable vector $\xi$ can contain deterministic variables and random variables as the ones subject to design. As far as the deterministic variables are concerned there is no ambiguity in the choice of variables as they refer to one number. In principle, the random variables can not be directly subject to design as they can take an infinite number of values. However, their probabilistic moments such as mean, variance, skewness and kurtosis can be subject to design. Although it is exciting to notice the possibility of having $\mu_\xi$ (the first moment) and higher moments as design variables, the reality is the higher moments are not in our control. The higher moments are simply mathematical a degrees of freedom that can be subject to optimization. Hypothetically, even if we give the control of choosing $\sigma_\xi$ (the second moment) to the optimizer, the optimizer would want to drive $\sigma_\xi$ to 0. This only implies that the designer should eliminate all uncertainties (i.e. the variability), but we are interested in OUU because the variability $\sigma_\xi$ is non zero and quantified to a number. Therefore, throughout this work when random variables are a part

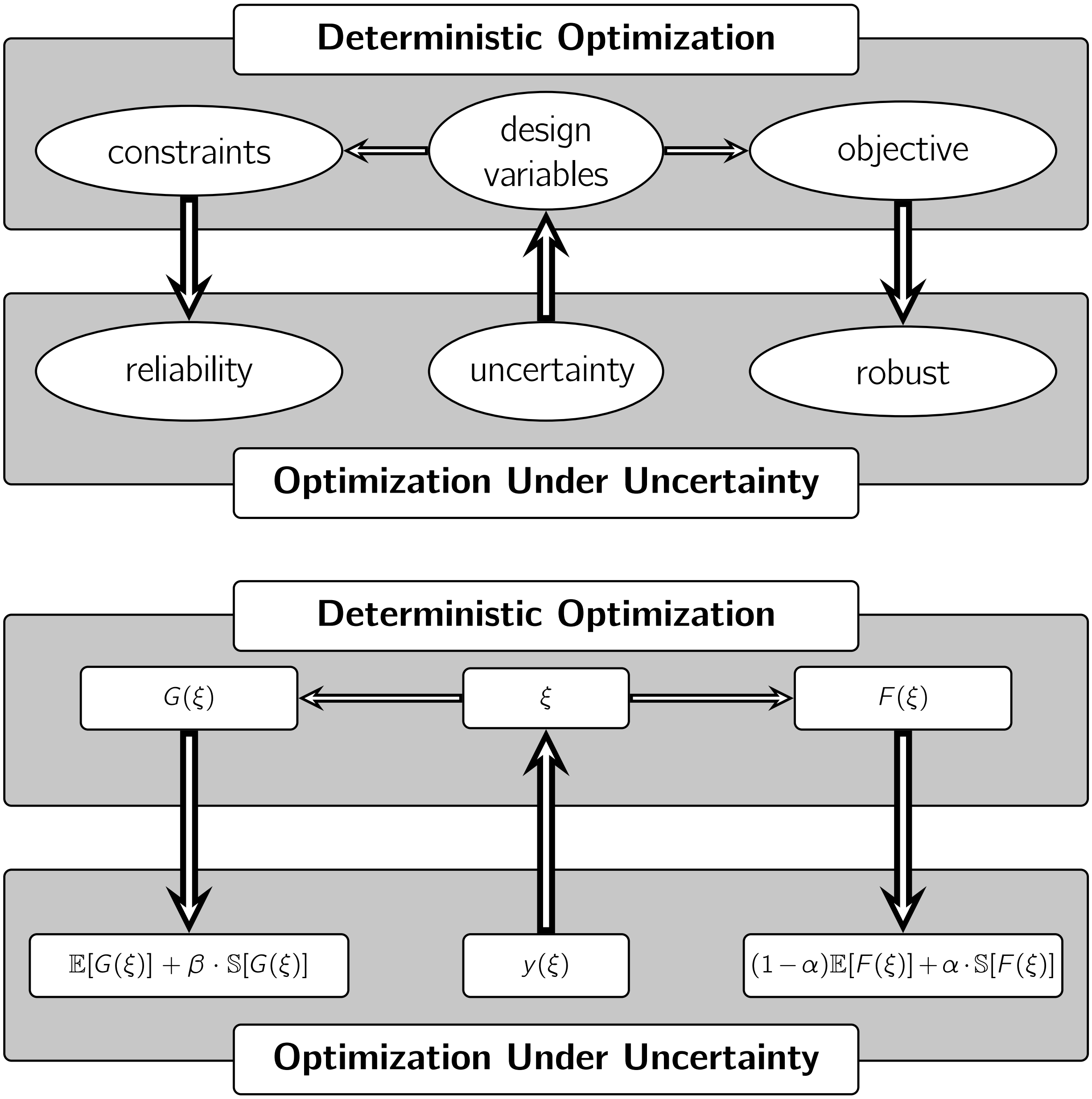

Deterministic Optimization
constraints
design variables
objective
reliability
uncertainty
robust
Optimization Under Uncertainty
Deterministic Optimization
$G(\xi)$
$\xi$
$F(\xi)$
$\mathbb{E}[G(\xi)] + \beta \cdot \mathbb{S}[G(\xi)]$
$y(\xi)$
$(1-\alpha)\mathbb{E}[F(\xi)] + \alpha \cdot \mathbb{S}[F(\xi)]$
Optimization Under Uncertainty

*Figure 2.10: Origination of robustness and reliability arguments from the space of deterministic optimization as soon as uncertainties are introduced.*

of design vector, only the first moment (mean) is subject to design.

$$\xi = [\ \underbrace{\xi_1, \xi_2,}_{\text{deterministic}}\ , \underbrace{y_1(\mu_{\xi_3}), y_2(\mu_{\xi_4})}_{\text{random}}] \tag{2.13}$$

**Objective Function:** The objective of minimizing the expected performance along with its variability can be stated mathematically as

$$\underset{\xi}{\text{minimize}} \quad \underbrace{(1-\alpha)\cdot\mathbb{E}[F(y(\xi))]}_{\text{expected performance}} \quad + \quad \underbrace{\alpha\cdot\mathbb{S}[F(y(\xi))]}_{\text{performance variability}} \tag{2.14}$$

with user-specified weights as $\alpha \in [0,1]$ that can be interpreted as parameter controlling robustness. Equation (2.14) can be viewed as a multiobjective optimization problem or as augmenting the mean objective with a weighted penalization using standard deviation. Some authors use two unconstrained weights $\alpha_1$ and $\alpha_2$ as well as variance in place of standard deviation [77].

**Constraint Function:** The designer may want to enforce directly that the probability of inequality constraint violation is less than a small number, for instance as $\mathbb{P}[G(y(\xi)) \leq 0] \geq b\%$. The probabilistic moments such as mean, variance and standard deviations are computationally easier to evaluate compared to the direct evaluation of probabilities. Therefore, an explicit enforcement of probabilities are difficult, where one can use implicit moment matching formulations (see Parkinson et al. [109], Du and Chen [105], Du and Chen [110]) to achieve the same effect. The probability statement can be restated as

$$\mathbb{P}[G(y(\xi)) \leq 0] \geq b\% \longrightarrow \quad \underbrace{\mathbb{E}[G(y(\xi))]}_{\text{location of constraint manifold}} \quad + \quad \underbrace{\beta\cdot\mathbb{S}[G(y(\xi))]}_{\text{shifting constraint manifold}} \quad \leq 0 \tag{2.15}$$

where $b$ is the desired probability and $\beta \in [0,\infty)$ can be interpreted as a parameter controlling reliability. The enforcement of equality constraints is rather tricky; see Rangavajhala et al. [111] for an overview of treatment of equality constraints. The simplest method is ensuring that the the optimal solution is sought along the manifold pertaining to mean of equality constraint, which is ensuring that the mean of the equality constraint is satisfied as

$$\mathbb{E}[H(y(\xi))] = 0. \tag{2.16}$$

**Probabilistic OUU Problem Statement:** Therefore, a general OUU problem embedding optimality, robustness and reliability design considerations can be stated as

$$\begin{aligned}
\underset{\xi}{\text{minimize}} \quad & (1-\alpha)\mathbb{E}[F(y(\xi))] + \alpha \cdot \mathbb{S}[F(y(\xi))] \\
\text{subject to} \quad & \mathbb{E}[G(y(\xi))] + \beta \cdot \mathbb{S}[G(y(\xi))] \leq 0 \\
& \mathbb{E}[H(y(\xi))] = 0 \\
\text{require} \quad & \frac{d\mathbb{E}[F(y(\xi))]}{d\xi}, \frac{d\mathbb{E}[G(y(\xi))]}{d\xi}, \frac{d\mathbb{E}[H(y(\xi))]}{d\xi} \\
& \frac{d\mathbb{S}[F(y(\xi))]}{d\xi}, \frac{d\mathbb{S}[G(y(\xi))]}{d\xi}
\end{aligned} \tag{2.17}$$

The optimization under uncertainty problem statement (2.17) helps to explain how the subjects reviewed in this chapter (temporal physics, gradient evaluation using adjoint method and uncertainty analysis) are connected to form the scope of the thesis:

- The **evaluation of functions of interest** $F(y(\xi))$, $G(y(\xi))$ and $H(y(\xi))$ falls within the realm of temporal physical analysis

- The **evaluation of probabilistic moments** of functions of interest $\mathbb{E}[F]$, $\mathbb{V}[F]$ and $\mathbb{S}[F]$ falls within the realm of uncertainty analysis

- The **evaluation of derivatives** of probabilistic moments of functions of interest $\frac{d\mathbb{E}[F]}{d\xi}$, $\frac{d\mathbb{V}[F]}{d\xi}$ and $\frac{d\mathbb{S}[F]}{d\xi}$ falls within the realm of sensitivity analysis

The proposed work of the thesis falls within the span of these three areas as illustrated schematically in Figure 2.11 with design applications to aeromechanical systems.

## 2.4 Specific Research Objectives

Figure 2.12 shows the mathematical process undertaking the proposed adjoint gradient enabled UQ-OUU framework. The static nonlinear physical analysis and corresponding linear sensitivity analysis solution techniques are well established in the current literature. The

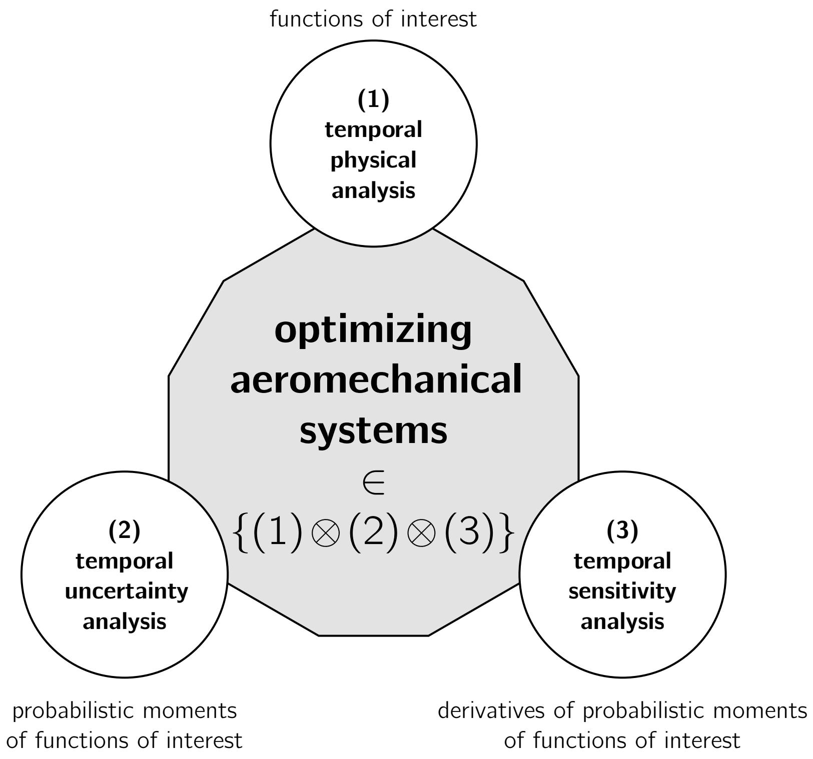


*Figure 2.11: A schematic diagram of the areas of mathematical developments required for adjoint-enabled UQ-OUU framework for time dependent systems.*

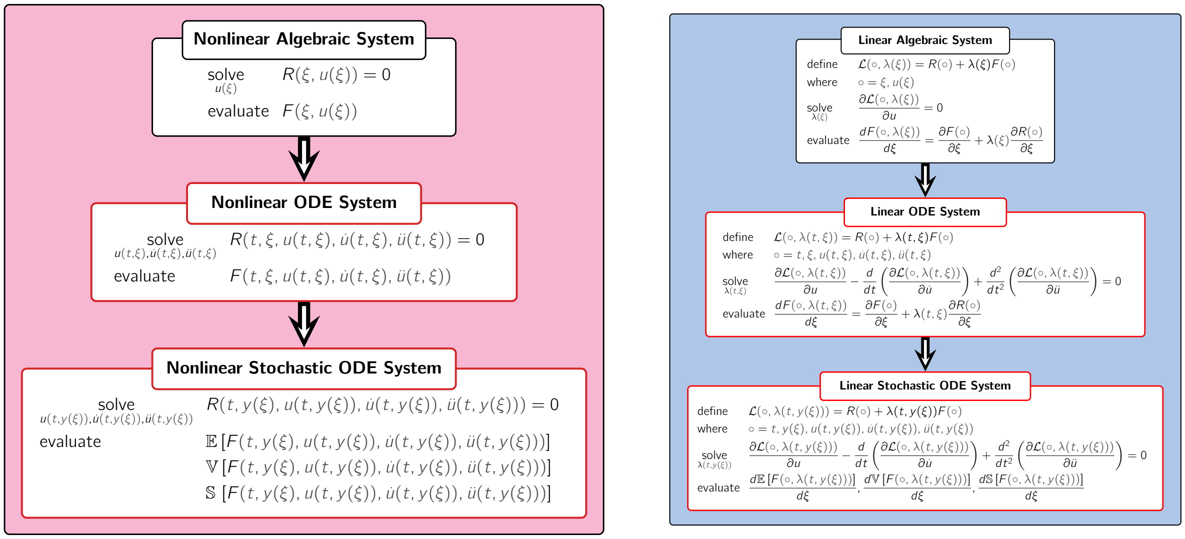
required new tasks for the proposed framework are highlighted in red. The detailed aspects of these high level goals are discussed in the remainder of this section along with pertaining novelties underlying this work.

### 2.4.1 Generalized Newton Method for Second-Order ODEs

In order to perform time-accurate analysis of physics, the first mathematical task is solving nonlinear ODE/DAE systems arising from flexible multibody dynamics and evaluating metrics of interest such as stress, strains, failure *etc*. This objective can be stated mathematically as:

$$\begin{aligned} &\underset{u(t,\xi),\dot{u}(t,\xi),\ddot{u}(t,\xi)}{\text{solve}} && R(t,\xi,u(t,\xi),\dot{u}(t,\xi),\ddot{u}(t,\xi)) = 0 \\ &\text{evaluate} && F(t,\xi,u(t,\xi),\dot{u}(t,\xi),\ddot{u}(t,\xi)). \end{aligned} \tag{2.18}$$

This extension of static analysis to time dependent analysis is approached from a philosophical standpoint of seeking to solve second-order DAE systems without converting them


Nonlinear Algebraic System
solve $_{u(\xi)}$ $R(\xi, u(\xi)) = 0$
evaluate $F(\xi, u(\xi))$
Nonlinear ODE System
solve $_{u(t,\xi),\dot{u}(t,\xi),\ddot{u}(t,\xi)}$ $R(t, \xi, u(t,\xi), \dot{u}(t,\xi), \ddot{u}(t,\xi)) = 0$
evaluate $F(t, \xi, u(t,\xi), \dot{u}(t,\xi), \ddot{u}(t,\xi))$
Nonlinear Stochastic ODE System
solve $_{u(t,y(\xi)),\dot{u}(t,y(\xi)),\ddot{u}(t,y(\xi))}$ $R(t, y(\xi), u(t,y(\xi)), \dot{u}(t,y(\xi)), \ddot{u}(t,y(\xi))) = 0$
evaluate $\mathbb{E}\left[F(t, y(\xi), u(t,y(\xi)), \dot{u}(t,y(\xi)), \ddot{u}(t,y(\xi)))\right]$
$\mathbb{V}\left[F(t, y(\xi), u(t,y(\xi)), \dot{u}(t,y(\xi)), \ddot{u}(t,y(\xi)))\right]$
$\mathbb{S}\left[F(t, y(\xi), u(t,y(\xi)), \dot{u}(t,y(\xi)), \ddot{u}(t,y(\xi)))\right]$
Linear Algebraic System
define $\mathcal{L}(\circ, \lambda(\xi)) = R(\circ) + \lambda(\xi)F(\circ)$
where $\circ = \xi, u(\xi)$
solve $_{\lambda(\xi)}$ $\frac{\partial \mathcal{L}(\circ, \lambda(\xi))}{\partial u} = 0$
evaluate $\frac{dF(\circ, \lambda(\xi))}{d\xi} = \frac{\partial F(\circ)}{\partial \xi} + \lambda(\xi)\frac{\partial R(\circ)}{\partial \xi}$
Linear ODE System
define $\mathcal{L}(\circ, \lambda(t,\xi)) = R(\circ) + \lambda(t,\xi)F(\circ)$
where $\circ = t, \xi, u(t,\xi), \dot{u}(t,\xi), \ddot{u}(t,\xi)$
solve $_{\lambda(t,\xi)}$ $\frac{\partial \mathcal{L}(\circ, \lambda(t,\xi))}{\partial u} - \frac{d}{dt}\left(\frac{\partial \mathcal{L}(\circ, \lambda(t,\xi))}{\partial \dot{u}}\right) + \frac{d^2}{dt^2}\left(\frac{\partial \mathcal{L}(\circ, \lambda(t,\xi))}{\partial \ddot{u}}\right) = 0$
evaluate $\frac{dF(\circ, \lambda(t,\xi))}{d\xi} = \frac{\partial F(\circ)}{\partial \xi} + \lambda(t,\xi)\frac{\partial R(\circ)}{\partial \xi}$
Linear Stochastic ODE System
define $\mathcal{L}(\circ, \lambda(t,y(\xi))) = R(\circ) + \lambda(t,y(\xi))F(\circ)$
where $\circ = t, y(\xi), u(t,y(\xi)), \dot{u}(t,y(\xi)), \ddot{u}(t,y(\xi))$
solve $_{\lambda(t,y(\xi))}$ $\frac{\partial \mathcal{L}(\circ, \lambda(t,y(\xi)))}{\partial u} - \frac{d}{dt}\left(\frac{\partial \mathcal{L}(\circ, \lambda(t,y(\xi)))}{\partial \dot{u}}\right) + \frac{d^2}{dt^2}\left(\frac{\partial \mathcal{L}(\circ, \lambda(t,y(\xi)))}{\partial \ddot{u}}\right) = 0$
evaluate $\frac{d\mathbb{E}\left[F(\circ, \lambda(t,y(\xi)))\right]}{d\xi}, \frac{d\mathbb{V}\left[F(\circ, \lambda(t,y(\xi)))\right]}{d\xi}, \frac{d\mathbb{S}\left[F(\circ, \lambda(t,y(\xi)))\right]}{d\xi}$

*(a) Temporal physical analysis*

*(b) Sensitivity analysis*

*Figure 2.12: An illustration of steps required to enable the integration of uncertainty quantification, temporal analysis and adjoint sensitivities (boxed in red).*

to first-order equations. Such an approach will enhance the body of techniques available for numerical solution of flexible multibody systems beyond Generalized-$\alpha$ methods. Other fields like chaotic dynamics can also benefit from this approach as their governing equations contain as high as sixth-order derivatives in time (see Chlouverakis and Sprott [20]), and one does not need an equivalent set of first-order equations to solve the system. From an implicit solution process perspective, this mainly necessitates the generalization of Newton–Raphson method. Therefore this goal reduces to the generalization of Newton method for implicit multistep and multistage time marching methods such as BDF, ABM and DIRK.

### 2.4.2 Implicit Analysis of Stochastic Time Dependent Systems

With the inclusion of uncertainties the DAEs/ODEs of flexible multibody system become stochastic DAEs/ODEs. The analysis problem can be stated mathematically as follows

$$\underset{u(t,y(\xi)),\dot{u}(t,y(\xi)),\ddot{u}(t,y(\xi))}{\text{solve}} \quad R(t, y(\xi), u(t, y(\xi)), \dot{u}(t, y(\xi)), \ddot{u}(t, y(\xi))) = 0. \tag{2.19}$$

Once the stochastic state fields $u(t, y(\xi))$, $\dot{u}(t, y(\xi))$ and $\ddot{u}(t, y(\xi))$ are determined, the probabilistic moments such as the mean $\mathbb{E}[F]$, variance $\mathbb{V}[F]$ and standard deviation $\mathbb{S}[F]$ can be evaluated. As Krenk and Gutirez [59] note, projection-based methods for problems involving nonlinearities have not yet reached a mature stage. This is partly due to the difficulty in deriving explicit stochastic equations in the presence of nonlinearities [72]. In this work we propose to form stochastic algebraic equations implicitly and circumvent the requirement for explicit derivation of stochastic equations, which is a hurdle in the easier adaptation of projection-based methods. When the governing deterministic PDEs take complex nonlinear coupled forms, the explicit derivation of stochastic equations (in algebraic form) may not be possible as pointed by Xiu [72]. We acknowledge this observation and also point out that in reality, the explicit equations are not necessary and can be made an implicit part of computational machinery seeking the solution to SPDEs. We find and show that the stochastic residuals and Jacobians can be assembled on the fly; thus explicit stochastic algebraic equations are not a necessity. For example, it is sufficient if one is able to form Jacobian-vector products implicitly to be able to solve a linear system $Ax = b$. Similarly, it is sufficient to have deterministic algebraic equations resulting from the spatial discretization method of choice, for being able to solve the corresponding stochastic problem. In this work we show the applicability of this guiding principle by demonstrating on problems ranging in complexities.

### 2.4.3 Time Dependent Discrete Adjoint Sensitivities

The availability of adjoint-derivatives allows one to perform optimization under uncertainty (OUU) efficiently, in order to produce engineering designs that are robust and reliable. This can be achieved in two parts discussed below.

- First we extend the static adjoint sensitivity analysis to time dependent adjoint sensitivity analysis for BDF, ABM, DIRK and Newmark implicit time marching methods. In the spirit of generality, we appeal to abstract descriptor form of governing equations (2.1) and functions of interest in the presentation of equations without making contextual arguments on flexible multibody dynamics. This approach will enable the use of derived set of discrete equations for a wide range of applied mathematical problems beyond flexible multibody dynamics.

- Next, we extend the deterministic time dependent adjoint sensitivity analysis into stochastic time dependent adjoint sensitivity analysis, by addressing the propagation of uncertainties through adjoint equations. The principle of reusability of deterministic implementations for stochastic physical analysis applies to stochastic sensitivity analysis as well. As a result, the stochastic adjoint equations are formed without need for explicitly setting them up.

### 2.5 Foundational Principles

The work of this thesis lies in the treatment of time domain and probabilistic domain in mathematical models and the development pertaining solution mechanisms to solve engineering design optimization problems of the form (2.17). In achieving this goal, we introduce/follow the following principles that permeate and guide the mathematical developments presented in this thesis:

1. In the treatment of time domain, the governing equations are kept in their natural second (higher) order form from theoretical formulation to numerical solution. Using this principle, we demonstrate the solution process using existing time marching methods such as BDF, ABM, DIRK and Newmark to solve the system directly in higher-order form.

2. In the treatment of probabilistic domain, we propose a principle of reusability of deterministic implementations and apply it not just for stochastic physical analysis but also for sensitivity analysis. We refer to this as semi-intrusive method for uncertainty propagation which is a synthesis of intrusiveness and non-intrusiveness.

3. Finally, being interested in the generality of mathematical developments and computer implementation, we use the principle of abstraction that maximizes the generality. This paves a perfect layout for programming solution techniques; indeed, abstraction is one of the key concepts of object-oriented programming languages. Abstraction is hiding unnecessary details and exposing only information that is relevant to the task at hand. For example, $R(t, \xi, u(t, \xi), \dot{u}(t, \xi)) = 0$ shall serve as an abstraction for:

    - $u(t) - \xi \dot{u}(t) = 0$ (a linear ODE)
    - $\cos(u(t)) \times \dot{u}(t) - \dfrac{\dot{u}(t)^3}{\xi} = 0$ (a nonlinear ODE)

# Part II

# Treatment of Time Domain in Physical Analysis and Sensitivity Analysis Problems

# CHAPTER 3

# SEMIANALYTICAL SENSITIVITIES FOR STATIONARY SYSTEMS

> *... the idea of enlarging reality by including "tentative" possibilities and then selecting one of these by the condition that it minimizes a certain quantity, seems to bring purpose to the flow of natural events.*
>
> *Cornelius Lanczos [1893–1974]*

**Introduction.** Let us use the term *zeroth-order systems* to denote physical processes that lack time derivative (or derivative in corresponding independent variable). The equations governing zeroth-order systems are thus algebraic in nature, as opposed to first and second-order processes modeled as differential equations. An example of zeroth-order system can be a body at rest responding instantaneously to an external stimulus modeled as an algebraic equation. In this chapter let us consider semianalytical methods for sensitivity analysis of systems modeled by algebraic equations. This will provide sufficient basis and intuition for the derivation of these equations in the context of second-order differential equations in Chapter 4.

## 3.1 Solution of Zeroth-Order Systems

Consider a system of nonlinear equations $R = R(q(\xi), \xi)$, where $\xi$ are design variables and $q = q(\xi)$ are the state variables which are an implicit function of the design variables $\xi$ via the governing nonlinear equations. As a general procedure, the state variables $q(\xi)$ are found by solving the nonlinear system $R = 0$ using an iterative scheme such as Newton-Raphson method. Although well established, the mathematical details of linearization and iteration are described next, for a self-contained discussion. First, we obtain a series expansion of the nonlinear function $R$ about the current iterate $q_k$:

$$R\left(q_k + \Delta q_k, \xi\right) = R\left(q_k, \xi\right) + \frac{\partial R}{\partial q}\left(q_k, \xi\right)\Delta q_k + \frac{1}{2}\frac{\partial^2 R}{\partial q^2}\left(q_k, \xi\right)\Delta {q_k}^2 + \ldots \leq \epsilon_R \tag{3.1}$$

which is required to be zero upto some tolerance, to consider the nonlinear system solved. Consider only upto the linear part of the expansion (3.1) (since we are interested in repeatedly solving linear systems) and rewrite as follows:

$$\frac{\partial R}{\partial q}\left(q_k, \xi\right)\Delta q_k = -R(q_k, \xi) \tag{3.2}$$

which is solved for $\Delta q_k$ using some suitable method for the solution of linear system. This enables us to find a new linearization point $q_{k+1} = q_k + \Delta q_k$. The linearization and linear system solve for update, are continued until some suitable criteria is satisfied. As a subtle but important detail, in the above iterative procedure the design variables $\xi$ are known and therefore it was not necessary to carry out an expansion of $R$ about $\xi$.

## 3.2 Methods for Obtaining First Derivative (Gradient)

We are interested in some abstract function $F = F(q(\xi), \xi)$, that is a function of the state variables that were just determined and design variables that are known. In many applications such as optimal design and optimal control, we require the first-order and second-order dependence of this function $F$ on the design variables $\xi$: the gradient $\dfrac{dF}{d\xi}$ and Hessian $\dfrac{d^2F}{d\xi^2}$.

### 3.2.1 Obtaining First Derivative (Gradient)

If the implicit dependence of $q$ on $\xi$ is known, then this merely amounts to assembling terms in chain rule of differentiation as:

$$\frac{dF}{d\xi} = \frac{\partial F}{\partial q}\underbrace{\frac{dq}{d\xi}}_{\text{implicit}} + \frac{\partial F}{\partial \xi} \tag{3.3}$$

However, in general, the implicit dependence $\frac{dq}{d\xi}$ is not known. Thus we need to resort to some sophisticated technique to obtain this information, in order to construct the derivative (3.3). Above all, we do not have a governing system of equations for $\frac{dq}{d\xi}$, *i.e.,* we lack a direct set of relations to solve. This motivates the need for development of such equations through mathematical techniques. For this purpose we define a new functional, called the Lagrangian as follows:

$$\mathcal{L}(q(\xi), \lambda(\xi), \xi) = F(q(\xi), \xi) + \lambda(\xi) R(q(\xi), \xi). \tag{3.4}$$

We have introduced a new unknown function $\lambda(\xi)$, referred to as the Lagrange multiplier, which is used to form a linear combination of the two functionals $F$ and $R$. The domain of the Lagrangian $\mathcal{L}$ is bigger than the domain of the function of interest $F$ and the governing physical equations $R$.

**Equivalence of $\mathcal{L}$ and $\mathcal{F}$** We shall first explore the conditions under which the new function (3.4) and the function of interest $F$ are identical. Noting that we solve for the nonlinear equations using Newton's method such that $R \leq \epsilon_R$. Therefore,

$$\mathcal{L}(q(\xi), \lambda(\xi), \xi) = F(q(\xi), \xi) + \lambda(\xi) \epsilon_R. \tag{3.5}$$

As long as $\epsilon_R \to 0$, it is trivial to see that one recovers the identity relation between between $\mathcal{L}$ and $F$. Therefore, the two functionals $\mathcal{L}$ and $\mathcal{F}$ are identical except for the existence of an additional dimension $\lambda$ in $\mathcal{L}$, and provided that the governing equations are solved to a tight tolerance.

**Differentiating the Lagrangian** Now we shall continue exploring the flexibility that the auxiliary variable $\lambda(\xi)$ offers to address the problem of unknown dependence $\frac{dq}{d\xi}$. Note that, ($i$) capturing the dependence or ($ii$) being orthogonal to the dependence are the solutions one can expect. Recall that the derivative of the original function is given by (3.3). Now, differentiating the Lagrangian (3.4) with respect to $\xi$, we get

$$\begin{aligned}\frac{d\mathcal{L}}{d\xi}(q(\xi),\lambda(\xi),\xi) &= \frac{dF}{d\xi}(q(\xi),\xi) + \frac{d}{d\xi}\left(\lambda(\xi)R(q(\xi),\xi)\right)\\ &= \frac{dF}{d\xi}(q(\xi),\xi) + \frac{d\lambda(\xi)}{d\xi}\cancelto{0}{R(q(\xi),\xi)} + \lambda(\xi)\frac{dR}{d\xi}(q(\xi),\xi) \\ &= \left(\frac{\partial F}{\partial q}\frac{dq}{d\xi} + \frac{\partial F}{\partial \xi}\right) + \lambda(\xi)\left(\frac{\partial R}{\partial q}\frac{dq}{d\xi} + \frac{\partial R}{\partial \xi}\right)\end{aligned} \tag{3.6}$$

We anticipate that $\frac{d\mathcal{L}}{d\xi} = \frac{dF}{d\xi}$, since $\mathcal{L}$ and $\mathcal{F}$ are identical. Therefore, the following inner product ought to vanish, for equivalence of derivatives:

$$\left\langle \lambda(\xi) \;\middle|\; \left(\frac{\partial R}{\partial q}\frac{dq}{d\xi} + \frac{\partial R}{\partial \xi}\right)\right\rangle = 0. \tag{3.7}$$

This relation is indeed the mathematical flexibility that the Lagrange Multipliers offer to solve the problem of unknown $q$ dependence of $\xi$.

### 3.2.2 A Direct Method

Out of the many possibilities where the inner product is zero, the trivial solution where:

$$\frac{\partial R}{\partial q}\frac{dq}{d\xi} + \frac{\partial R}{\partial \xi} = 0 \tag{3.8}$$

is known as the method of *direct sensitivities.* This linear system can be solved for the unknown $\frac{dq}{dx}$, and can be used to evaluate the total derivative of the Lagrangian which is identical to the total derivative of the functional when $R \leq \epsilon_R \approx 0$. This is called the *direct method* as we determine the implicit dependence directly by solving a linear system of equations.

### 3.2.3 An Indirect (Adjoint) Method

We shall now explore what other options we have from (3.6). Suppose if we regroup terms in (3.6) as follows:

$$\frac{d\mathcal{L}}{d\xi} = \left(\frac{\partial F}{\partial q} + \lambda\frac{\partial R}{\partial q}\right)\frac{dq}{d\xi} + \left(\frac{\partial F}{\partial \xi} + \lambda\frac{\partial R}{\partial \xi}\right) \tag{3.9}$$

By the same token as before, the following condition is another set of mathematical possibilities that we get by introducing Lagrange multipliers:

$$\left\langle \left(\frac{\partial F}{\partial q} + \lambda\frac{\partial R}{\partial q}\right) \middle| \frac{dq}{d\xi} \right\rangle = 0 \tag{3.10}$$

The trivial case where

$$\frac{\partial F}{\partial q} + \lambda\frac{\partial R}{\partial q} = 0 \tag{3.11}$$

yields us the condition that is used to solve for the Lagrange multiplier $\lambda$, and is referred to as the *adjoint method*.

### 3.2.4 An Illustrative Example

We consider a system governed by linear algebraic equation. A spring with stiffness constant $\xi$ responds to an external stimulus $b$ and displaces by an amount $q$, is modeled as:

$$R := R(q(\xi), \xi) = \xi q - b$$

Let us define a quantity of interest

$$F := F(q(\xi), \xi) = \frac{1}{2}\xi q^2.$$

Also, let the spring stiffness $\xi$ be the variable subject to design. The known partial derivatives are:

$$\frac{\partial F}{\partial \xi} = \frac{1}{2}q^2,\ \frac{\partial R}{\partial \xi} = q,\ \frac{\partial F}{\partial q} = \xi q \text{ and } \frac{\partial R}{\partial q} = \xi.$$

**1. Exact Derivative:** For this simple system the implicit derivative is known as

$$\frac{dq}{d\xi} = -\frac{b}{\xi^2} = -\frac{q}{\xi}$$

which can be used directly assemble the exact analytical derivative using chain rule as follows

$$\frac{dF}{d\xi} = \frac{\partial F}{\partial \xi} + \frac{\partial F}{\partial q}\frac{dq}{d\xi} = \frac{1}{2}q^2 + (\xi q)\left(-\frac{q}{\xi}\right) = -\frac{1}{2}q^2.$$

Often, the implicit dependence is not known and thus we work with the assumption that it is not readily available.

**2. Direct Method:** In the case of the direct method, the implicit derivative is directly computed by means of solving the linear system

$$\frac{dq}{d\xi} = -\frac{\partial R}{\partial \xi}\Bigg/\frac{\partial R}{\partial q} = -\frac{q}{\xi}.$$

This implicit derivative can be used to assemble the first derivative as

$$\frac{d\mathcal{L}}{d\xi} = \frac{\partial F}{\partial \xi} + \frac{\partial F}{\partial q}\frac{dq}{d\xi} = \frac{1}{2}q^2 + (\xi q)\left(-\frac{q}{\xi}\right) = -\frac{1}{2}q^2.$$

**3. Adjoint Method:** Using the adjoint method we find the adjoint variable by solving a linear system

$$\lambda = -\frac{\partial F}{\partial q}\Bigg/\frac{\partial R}{\partial q} = -q,$$

which can be used to assemble the required derivative as

$$\frac{d\mathcal{L}}{d\xi} = \frac{\partial F}{\partial \xi} + \lambda\frac{\partial R}{\partial \xi} = \frac{1}{2}q^2 + (-q)(q) = -\frac{1}{2}q^2.$$

**Summary of obtaining first derivatives.** Using abstractions of governing equations and functionals of interest, we derived two established state-of-the-art approaches for obtaining the derivatives of functionals with respect to the independent parameter and outlined other avenues for computing the derivatives. It is noted that in all cases, the linear nature of Lagrangian results in linear system solves such as (3.8) and (3.11). We will explore the time dependent adjoint sensitivity analysis details using similar abstractions and principles in Chapter 4.

# CHAPTER 4
# TIME MARCHING AND DISCRETE-ADJOINT FOR SECOND-ORDER SYSTEMS

**Introduction.** This chapter presents numerical solution methods for the governing equations of flexible multibody dynamics using implicit time marching methods, as well as the development of associated discrete-adjoint equations. The implicit time marching methods considered are:

1. Newmark method [11, 12] (single-step),
2. Backward Difference Formulas (BDF) [14, 15] (multistep),
3. Adams–Bashforth–Moulton method (ABM) [21, 22] (multistep),
4. Diagonally Implicit Runge–Kutta (DIRK) [24, 25] (multistage).

Starting from a given set of initial conditions $q_0$ and $\dot{q}_0$, these time marching schemes use the state variable values from previous time steps to evaluate the subsequent values of state variables. The subscript, $k$, on the state variables refer to the corresponding time parameter value, $t_k$. In this work, a constant step size, $h = t_k - t_{k-1}$, is employed for time marching. The scalar coefficients of time marching denoted as $\alpha$, $\beta$ and $\gamma$ are used for forming linear approximations of the state variables at each time step. These coefficients are derived based on desired order of accuracy and stability requirements.

## 4.1 Governing Equations of Motion and Continuous Adjoint

In this section, we use variational principles to derive the governing equations for flexible multibody systems and continuous adjoint-based sensitivities. Our approach is to operate on a broader paradigm, until the contextual arguments are used to recover Euler-Lagrange and adjoint equations.

### 4.1.1 Variational Principle

We begin with Lagrangians that are function of the state variables and their first as well as second time derivatives. It is easy to see that $q = q(t)$ denotes a set of curves (paths, trajectories or functions) that the components of the state variables trace over time, and there are as many scalar curves as the number of state variables. Similarly, a Lagrangian can be seen as a curve in $3m+1$ dimensional space spanned by $q, \dot{q}$, $\ddot{q}$ and $t$. The length $\mathcal{A}$ of the Lagrangian curve within the time interval $[t_0, t_f]$ is represented as a line integral along the curve

$$\mathcal{A}[\ddot{q}, \dot{q}, q] = \int_{t_0}^{t_f} \mathcal{L}(\ddot{q}, \dot{q}, q, t)\; dt. \tag{4.1}$$

Consider the following variations in state trajectories

$$\begin{aligned} q(t) &\to q(t) + \delta q(t), \\ \dot{q}(t) &\to \dot{q}(t) + \delta \dot{q}(t), \\ \ddot{q}(t) &\to \ddot{q}(t) + \delta \ddot{q}(t), \end{aligned} \tag{4.2}$$

where $\delta q(t), \delta\dot{q}(t)$ and $\delta\ddot{q}(t)$ are arbitrary changes to the corresponding original trajectories, that vanish at the end points $t_0$ and $t_f$. Thus, the varied paths have the boundary conditions that

$$\delta q(t_0) = \delta q(t_f) = \delta\dot{q}(t_0) = \delta\dot{q}(t_f) = \delta\ddot{q}(t_0) = \delta\ddot{q}(t_f) = 0. \tag{4.3}$$

In order to study the effect of these varied paths on the length $\mathcal{A}$, we use Taylor's series expansion of the Lagrangian upto first-order terms as

$$\mathcal{L}(\ddot{q}+\delta\ddot{q}, \dot{q}+\delta\dot{q}, q+\delta q, t) \approx \mathcal{L}(\ddot{q}, \dot{q}, q, t) + \frac{\partial \mathcal{L}(\ddot{q}, \dot{q}, q, t)}{\partial q}\delta q(t) + \frac{\partial \mathcal{L}(\ddot{q}, \dot{q}, q, t)}{\partial \dot{q}}\delta\dot{q}(t) + \frac{\partial \mathcal{L}(\ddot{q}, \dot{q}, q, t)}{\partial \ddot{q}}\delta\ddot{q}(t), \tag{4.4}$$

and similarly the functional

$$\mathcal{A}[\ddot{q}+\delta\ddot{q},\dot{q}+\delta\dot{q},q+\delta q] \approx \mathcal{A}[\ddot{q},\dot{q},q]+\delta\mathcal{A}[\ddot{q},\dot{q},q]. \tag{4.5}$$

The higher-order terms in the expansion are omitted to keep the analysis simpler, although with an intent of mitigating the incurred truncation error by taking small perturbations in the neighborhood of the reference values of expansion. The first variation of the functional $\mathcal{A}$ is

$$\begin{aligned}\delta\mathcal{A} &= \int_{t_0}^{t_f} \mathcal{L}(\ddot{q}+\delta\ddot{q},\dot{q}+\delta\dot{q},q+\delta q,t)\ dt - \int_{t_0}^{t_f} \mathcal{L}(\ddot{q},\dot{q},q,t)\ dt \\ &= \int_{t_0}^{t_f} \left(\frac{\partial\mathcal{L}}{\partial\ddot{q}}\delta\ddot{q}+\frac{\partial\mathcal{L}}{\partial\dot{q}}\delta\dot{q}+\frac{\partial\mathcal{L}}{\partial q}\delta q\right) dt \\ &= \int_{t_0}^{t_f} \left[\frac{d^2}{dt^2}\left(\frac{\partial\mathcal{L}}{\partial\ddot{q}}\right)-\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{q}}\right)+\frac{\partial\mathcal{L}}{\partial q}\right]\delta q\ dt.\end{aligned} \tag{4.6}$$

Note that *integration by parts* and boundary conditions (4.3) are used in arriving at the above result. We are interested in finding the *critical curves* $q(t), \dot{q}(t)$ and $\ddot{q}(t)$ of the functional $\mathcal{A}$ that have the property of rendering the length $\mathcal{A}$ unchanged to its first-order expansion in Taylor's series, when its arguments are perturbed to $q(t)+\delta q(t)$, *etc.* In other words, we invoke the Hamilton's principle that the *first variation* of functional, $\delta\mathcal{A}$, vanishes to zero for critical trajectories yielding

$$\frac{\partial\mathcal{L}}{\partial q}-\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{q}}\right)+\frac{d^2}{dt^2}\left(\frac{\partial\mathcal{L}}{\partial\ddot{q}}\right)=0. \tag{4.7}$$

### 4.1.2 Euler-Lagrange Equations

The Lagrangian functions pertaining to flexible-multibody dynamic systems are known to be functions of the state variables and their first time derivatives, *i.e.,* $\mathcal{L}=\mathcal{L}(\dot{q},q,t)$. By specializing (4.7) to such Lagrangians that are a function of kinetic and potential energies of the system, we recover the much revered *Euler-Lagrange equations*

$$\frac{\partial \mathcal{L}}{\partial q} - \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) = 0 \tag{4.8}$$

The alternate form of Euler–Lagrange equations is obtained as

$$\frac{\partial \mathcal{L}}{\partial q} - \frac{\partial \mathcal{L}}{\partial q}\dot{q} - \frac{\partial \mathcal{L}}{\partial \dot{q}}\ddot{q} - \frac{\partial}{\partial t}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) \tag{4.9}$$

The above second-order differential equation (4.9) is represented in abstract descriptor form as $R(t, \xi, \ddot{q}, \dot{q}, q)$. The unknown states variables and their time derivatives can be determined using numerical time marching from initial conditions specified at $t = t_0$. This process is commonly referred to as *forward/state solution mode* in the literature.

### 4.1.3 Continuous Adjoint

We now turn our attention to the problem of finding the sensitivities of a functional of interest with respect to design variables $\xi$, using Hamilton's principle. To derive the equations, we now form a Lagrangian by augmenting the functional of interest, with the inner product of the adjoint variables $\lambda = \lambda(t)$ and governing equations as follows

$$\mathcal{L}(t, \xi, \ddot{q}, \dot{q}, q) = F(t, \xi, \ddot{q}, \dot{q}, q) + \lambda(t, \xi)^T R(t, \xi, \ddot{q}, \dot{q}, q), \tag{4.10}$$

The curve for $\lambda(t)$ is arbitrary except at $t = t_f$ where it vanishes *i.e.,* $\lambda(t_f) = 0$. The Lagrangian (4.10) can be evaluated only after the state variables are determined from the *forward solution mode* and we only know the value of $\lambda$ at the final time $t_f$. Therefore, the solution to the problem of determining the unknown adjoint variables starts at the final time $t_f$ and marching backwards in time towards the initial time $t_0$. This process is referred to as the *reverse/adjoint solution mode* in the literature. The reversal of time coordinate amounts to transposition of linear algebra objects and flipping of signs of odd-time derivatives. It is convenient to introduce a dummy time variable $\tau$ whose coordinate direction is reversed. The result is the following governing equations for the adjoint problem

$$\frac{\partial \mathcal{L}^T}{\partial q} + \frac{d}{d\tau}\left(\frac{\partial \mathcal{L}^T}{\partial \dot{q}}\right) + \frac{d^2}{d\tau^2}\left(\frac{\partial \mathcal{L}^T}{\partial \ddot{q}}\right) = 0. \tag{4.11}$$

Using the definition of adjoint Lagrangian (4.10), we get

$$\frac{\partial F}{\partial q}^T + \frac{\partial R}{\partial q}^T \lambda + \frac{d}{d\tau}\left(\frac{\partial F}{\partial \dot{q}}^T + \frac{\partial R}{\partial \dot{q}}^T \lambda\right) + \frac{d^2}{d\tau^2}\left(\frac{\partial F}{\partial \ddot{q}}^T + \frac{\partial R}{\partial \ddot{q}}^T \lambda\right) = 0. \tag{4.12}$$

### 4.1.4 Governing Equations of Motion

The equations governing the motion of flexible multibody systems can be derived using a number of different methods [4, 112]. This work employs an approach based on the constrained Euler–Lagrange equations that leads to a system of differential algebraic equations (DAEs). The system of DAEs consists of both a set of differential equations and a set of algebraic constraints that restricts the kinematics using Lagrange multipliers. One advantage of using the Euler–Lagrange equations is that they can be numerically verified for consistency with the kinetic and potential energy expressions and the constraint equations using finite-difference or complex-step methods. The Lagrangian for the equations of motion is defined as

$$\mathcal{L}(\dot{w}, w) \triangleq T(\dot{w}, w) - V(w) \tag{4.13}$$

where $w$ is a vector that contains the displacements and Euler parameters for rotation matrix parametrization, and $T(\dot{w}, w)$ and $V(w)$ are the kinetic and potential energy of the system, respectively. The kinetic energy and potential energies are computed as integrals over each finite element [4, 113, 114]. In this work, the kinematics of the flexible bodies are restricted through a set of holonomic constraints of the form $g(w) = 0$, where the extension to nonholonomic constraints is straightforward. The Jacobian of the kinematic constraints is $A = \partial g/\partial w$. With these definitions, the governing equations of motion in second-order descriptor form are

$$R(t,\xi,\ddot{q},\dot{q},q) \triangleq \begin{bmatrix} \dfrac{d}{dt}\left(\dfrac{\partial \mathcal{L}}{\partial \dot{w}}\right) - \dfrac{\partial \mathcal{L}}{\partial w} - A^T\mu \\ g(w) \end{bmatrix} = 0. \tag{4.14}$$

Here the vector $q = (w,\ \mu)$ includes both the degrees of freedom $w$ and the Lagrange multipliers $\mu$. Note that the vector of design variables, $\xi$, is included to reflect the dependence of the system of equations on design variables. In the following sections, it will be necessary to compute the Jacobian of the governing equations with respect to the state variables and their derivatives. These Jacobian matrices always appear as a linear combinations of the form

$$J = \overline{\gamma}\frac{\partial R}{\partial \ddot{q}} + \overline{\beta}\frac{\partial R}{\partial \dot{q}} + \overline{\alpha}\frac{\partial R}{\partial q},$$

where $\overline{\alpha}$, $\overline{\beta}$, and $\overline{\gamma}$ are scalar coefficients. The descriptor form (4.14) provides the basis for different element types implemented in the framework based on the finite element method. The elements within the framework, at present, consist of rigid bodies, flexible quadratic beam elements employing a Timoshenko beam formulation, and flexible bi-quadratic shell elements employing a Reissner–Mindlin formulation. To avoid shear locking, the beam and shell elements employ a mixed interpolation of tensorial components (MITC) formulation [113, 114]. In addition, kinematic constraints are implemented within the same element hierarchy, including the lower kinematic pairs [4, 115].

## 4.2 Newmark Method

The Newmark family of integrators are single-step methods that use state variable values and their time derivatives from the previous step. The pioneering work of this method were by Fox and Goodwin [11] and Newmark [12]. This method was originally developed for the numerical solution of problems in structural dynamics including linear elastic studies, dynamic loading and vibrations due to earthquake. It has subsequently found applications in flexible multibody dynamics over the years. The order of accuracy, $p$, of the Newmark scheme depends on the choice of the coefficients, $\beta$ and $\gamma$, as listed in Table 4.1. The Newmark method is now a part of Generalized-$\alpha$ class of methods presented by Chung and Hilbert [13].

*Table 4.1: The coefficients of Newmark family of methods and their corresponding orders of accuracy.*

| Method | $\beta$ | $\gamma$ | Order |
|---|---|---|---|
| Implicit Fox-Goodwin [11] | 1/12 | 1/2 | 3 |
| Implicit linear acceleration | 1/6 | 1/2 | 2 |
| Implicit average constant acceleration | 1/4 | 1/2 | 2 |
| Implicit central difference | 0 | 1/2 | 2 |
| Explicit | 0 | 0 | 1 |

### 4.2.1 Solution of the State Variables

#### *4.2.1.1 State Approximation Hypothesis*

The primary unknowns of the Newmark method are the second time derivatives of the state variables $\ddot{q}_k$ at each time step $k$. The $p$-th order state approximations are:

$$\begin{aligned} \dot{q}_k &= \dot{q}_{k-1} + (1-\gamma)h\ddot{q}_{k-1} + \gamma h\ddot{q}_k + \mathcal{O}(h^p) \\ q_k &= q_{k-1} + h\dot{q}_{k-1} + \frac{1-2\beta}{2}h^2\ddot{q}_{k-1} + \beta h^2\ddot{q}_k + \mathcal{O}(h^p). \end{aligned} \tag{4.15}$$

The state approximations (4.15) are simply weighted linear combinations of state vector functions as illustrated in Figure 4.1.

#### *4.2.1.2 Solution of the Nonlinear System*

The nonlinear governing equations at each step, $R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k) = 0$, are linearized with respect to $\ddot{q}_k$ as follows

$$\left[\frac{\partial R_k}{\partial \ddot{q}_k} + \gamma h\frac{\partial R_k}{\partial \dot{q}_k} + \beta h^2\frac{\partial R_k}{\partial q_k}\right]\Delta\ddot{q}_k = -R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k). \tag{4.16}$$

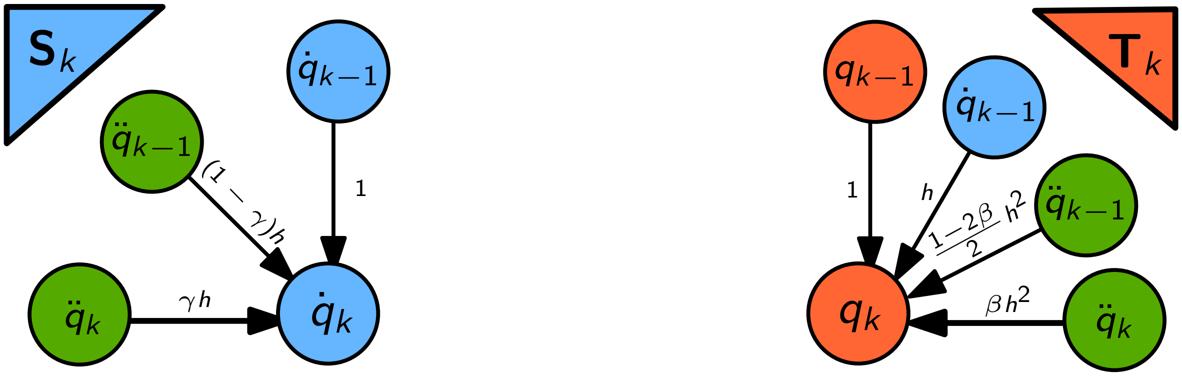


*Figure 4.1: Weighted linear combination of states yielding the velocity (left) and position (right) states for Newmark method using state approximation hypothesis $S_k$ and $T_k$.*

The state variables and its first time derivatives (velocities) are approximated using Equation (4.15), for an estimated value of the acceleration state variables (second time derivatives), at a known time, $t_k$. The linear system (4.16) is then solved for the primary update $\Delta\ddot{q}_k^n$ (incremental accelerations), at each iteration, $n$, of the nonlinear solution. The secondary and tertiary updates required for the state variables and their first time derivatives are readily obtained by scaling the acceleration update using the Newmark coefficients. The resulting update formulas are

$$\begin{aligned} \ddot{q}_k^{n+1} &= \ddot{q}_k^n + \Delta\ddot{q}_k^n, \\ \dot{q}_k^{n+1} &= \dot{q}_k^n + \gamma h \Delta\ddot{q}_k^n, \\ q_k^{n+1} &= q_k^n + \beta h^2 \Delta\ddot{q}_k^n. \end{aligned} \tag{4.17}$$

The iterative updates to the state variables and their derivatives are continued until the governing equations are solved to the required tolerance. The accuracy of adjoint derivatives rely on the accuracy of the solution of the governing equations. Therefore, it is important that the discrete nonlinear system (4.16) is solved to a tight tolerance.

### 4.2.2 Solution of the Adjoint Variables

#### *4.2.2.1 Formation of the Lagrangian*

The state approximations of the Newmark time marching scheme given in Equation (4.15) are reformulated as following residuals

$$\begin{aligned} S_k &= \dot{q}_{k-1} + (1-\gamma)h\ddot{q}_{k-1} + \gamma h\ddot{q}_k - \dot{q}_k, \\ T_k &= q_{k-1} + h\dot{q}_{k-1} + \frac{1-2\beta}{2}h^2\ddot{q}_{k-1} + \beta h^2\ddot{q}_k - q_k. \end{aligned} \tag{4.18}$$

The adjoint variables, $\lambda_k$, $\psi_k$ and $\phi_k$ are introduced as respective unknown weights, to the governing equations, $R_k$, the state approximation equations, $S_k$, and $T_k$, arising from the Newmark scheme, for each time step, $k$. The Lagrangian is formed as the following linear combination of functions from the span of $\ddot{q}(t,\xi) \otimes \dot{q}(t,\xi) \otimes q(t,\xi) \otimes \xi \otimes t$:

$$\mathcal{L} = \sum_{k=0}^{N} hF_k + \sum_{k=0}^{N} h\lambda_k^T R_k + \sum_{k=0}^{N} \psi_k^T S_k + \sum_{k=0}^{N} \phi_k^T T_k. \tag{4.19}$$

The central idea is to represent the functional, $F_k$, as a linear combination of the other equations, and identify trajectories $\phi(t,\xi)$, $\psi(t,\xi)$ and $\lambda(t,\xi)$ that are invariant to perturbations in $q(t,\xi)$, $\dot{q}(t,\xi)$ and $\ddot{q}(t,\xi)$, respectively. An illustration of the formation of Lagrangian for Newmark method is shown in Figure 4.2.

#### *4.2.2.2 The Adjoint Equations*

The system of equations to solve for the adjoint variables at each time step is obtained from the stationary points of the Lagrangian with respect the state variables and their time derivatives. The number of partial derivative terms that exist in the adjoint system of equations, can be graphically determined from Figure 4.2, based on the occurrences of $q_k$, $\dot{q}_k$ and $\ddot{q}_k$ as inputs to equations $F$, $R$, $S$ and $T$, at different time steps; see Table 4.2 for a list of number of terms in the adjoint system of equations.

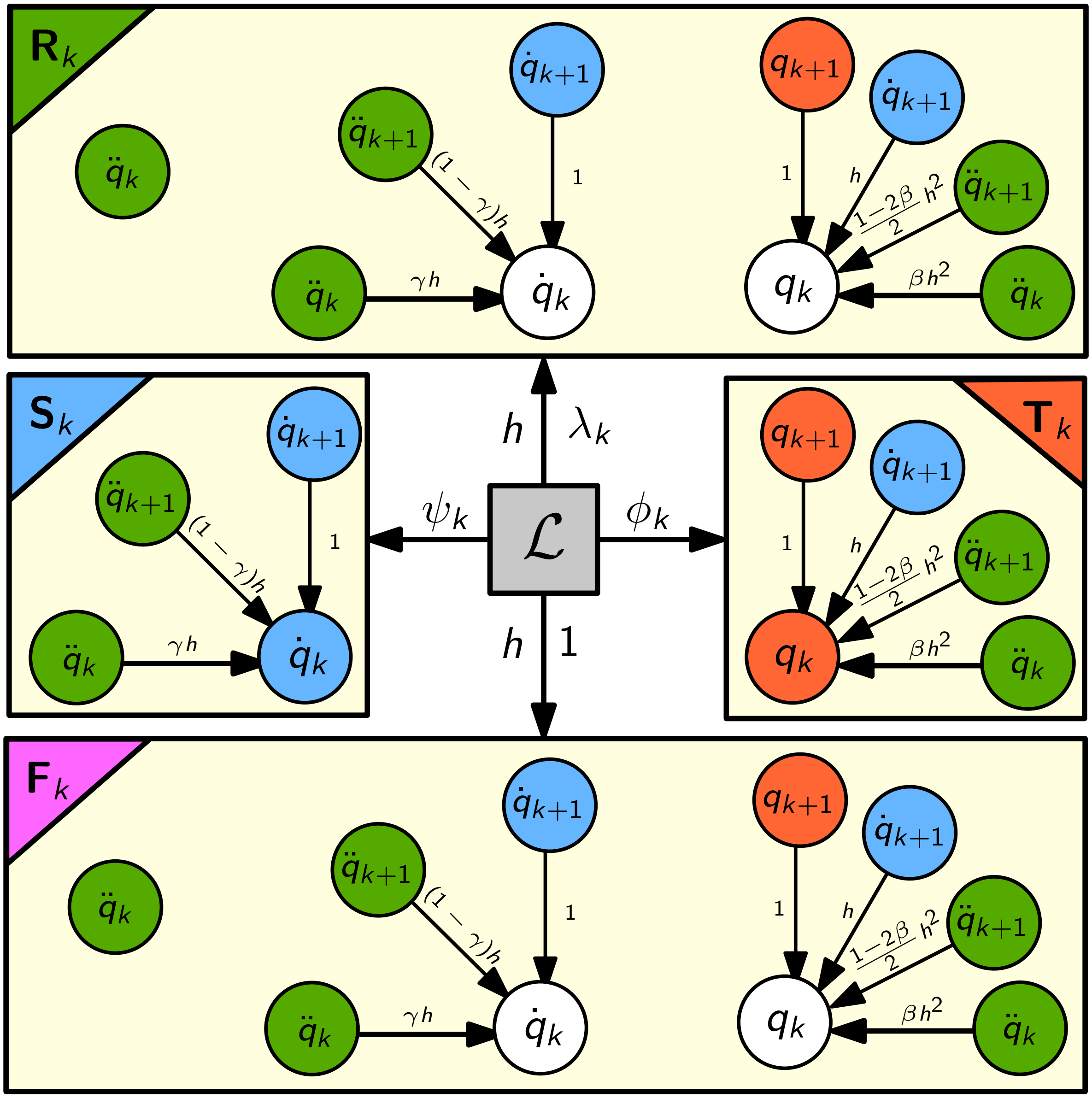


*Figure 4.2: The weighted linear combinations of equations with corresponding adjoint variables forming the Lagrangian for Newmark method.*

**1. Equation for $\phi_k$:** Setting $\partial\mathcal{L}/\partial q_k = 0$ yields

$$\frac{\partial T_k}{\partial q_k}^T \phi_k + \frac{\partial T_{k+1}}{\partial q_k}^T \phi_{k+1} + h\frac{\partial R_{k+1}}{\partial q_k}^T \lambda_{k+1} + h\frac{\partial F_{k+1}}{\partial q_k}^T = 0. \tag{4.20}$$

*Table 4.2: The number of terms in the adjoint system of equations for Newmark family of integrators.*

| | $T$ | $S$ | $R$ | F | Total |
|---|---|---|---|---|---|
| $\phi_k$ | 2 | | 1 | 1 | 4 |
| $\psi_k$ | 1 | 2 | 2 | 2 | 7 |
| $\lambda_k$ | 2 | 2 | 5 | 5 | 14 |

Further simplifications result in

$$\phi_k = \phi_{k+1} + h \left[ \frac{\partial R_{k+1}}{\partial q_{k+1}} \right]^T \lambda_{k+1} + h \left\{ \frac{\partial F_{k+1}}{\partial q_{k+1}} \right\}^T . \tag{4.21}$$

It can be noticed that the four terms of Equation (4.21) correspond to the occurrences of the primal variable $q_k$ as inputs to equations during forward solution history.

<u>**2. Equation for $\psi_k$:**</u> Using $\partial \mathcal{L} / \partial \dot{q}_k = 0$:

$${\frac{\partial S_k}{\partial \dot{q}_k}}^T \psi_k + {\frac{\partial S_{k+1}}{\partial \dot{q}_k}}^T \psi_{k+1} + {\frac{\partial T_{k+1}}{\partial \dot{q}_k}}^T \phi_{k+1} + h {\frac{\partial R_{k+1}}{\partial \dot{q}_k}}^T \lambda_{k+1} + h {\frac{\partial F_{k+1}}{\partial \dot{q}_k}}^T = 0. \tag{4.22}$$

This simplifies to

$$\psi_k = \psi_{k+1} + h\phi_{k+1} + h \left[ \frac{\partial R_{k+1}}{\partial \dot{q}_{k+1}} + h \frac{\partial R_{k+1}}{\partial q_{k+1}} \right]^T \lambda_{k+1} + h \left\{ \frac{\partial F_{k+1}}{\partial \dot{q}_{k+1}} + h \frac{\partial F_{k+1}}{\partial q_{k+1}} \right\}^T . \tag{4.23}$$

The seven terms in Equation (4.23) represent and accumulate contributions from residuals/equations that were affected by $\dot{q}_k$ during the forward mode.

<u>**3. Equation for $\lambda_k$:**</u> Setting $\partial \mathcal{L} / \partial \ddot{q}_k = 0$ yields the equation to solve for the adjoint variable, $\lambda_k$,

$$h \sum_{i=0}^{1} {\frac{\partial R_{k+i}}{\partial \ddot{q}_k}}^T \lambda_{k+i} + h \sum_{i=0}^{1} {\frac{\partial F_{k+i}}{\partial \ddot{q}_k}}^T + \sum_{i=0}^{1} {\frac{\partial S_{k+i}}{\partial \ddot{q}_k}}^T \psi_{k+i} + \sum_{i=0}^{1} {\frac{\partial T_{k+i}}{\partial \ddot{q}_k}}^T \phi_{k+i} = 0. \tag{4.24}$$

Rearranging the terms results in the following linear system for $\lambda_k$

$$\begin{aligned}\left[\frac{\partial R_k}{\partial \ddot{q}_k} + \gamma h \frac{\partial R_k}{\partial \dot{q}_k} + \beta h^2 \frac{\partial R_k}{\partial q_k}\right]^T \lambda_k = &- \left\{\frac{\partial F_k}{\partial \ddot{q}_k} + \gamma h \frac{\partial F_k}{\partial \dot{q}_k} + \beta h^2 \frac{\partial F_k}{\partial q_k}\right\}^T \\ &- \frac{1}{h}\left\{\gamma h \psi_k + \beta h^2 \phi_k\right\}^T \\ &- \left[(1-\gamma) h \frac{\partial R_{k+1}}{\partial \dot{q}_{k+1}} + \frac{1-2\beta}{2} h^2 \frac{\partial R_{k+1}}{\partial q_{k+1}}\right]^T \lambda_{k+1} \qquad (4.25) \\ &- \left\{(1-\gamma) h \frac{\partial F_{k+1}}{\partial \dot{q}_{k+1}} + \frac{1-2\beta}{2} h^2 \frac{\partial F_{k+1}}{\partial q_{k+1}}\right\}^T \\ &- \frac{1}{h}\left\{(1-\gamma) h \psi_{k+1} + \frac{1-2\beta}{2} h^2 \phi_{k+1}\right\}^T .\end{aligned}$$

The fourteen terms in (4.25) can be graphically interpreted in Figure 4.2, as occurrences of the primal state variable $\ddot{q}_k$ as inputs argument to equations in the time history.

#### *4.2.2.3 Evaluation of Total Derivative:*

The determination of adjoint variables $\lambda(t, \xi)$, $\phi(t, \xi)$ and $\psi(t, \xi)$ allows evaluating the total derivative of the function of interest with respect to the design variables $\xi$ as

$$\frac{df}{d\xi} = \sum_{k=0}^{N} h \frac{\partial F_k}{\partial \xi} + \sum_{k=0}^{N} h \lambda_k^T \frac{\partial R_k}{\partial \xi} + \sum_{k=0}^{N} \psi_k^T \frac{\partial S_k}{\partial \xi} + \sum_{k=0}^{N} \phi_k^T \frac{\partial T_k}{\partial \xi}. \qquad (4.26)$$

Since, the state approximation equations $S_k$ and $T_k$ are independent of the design variables, $\xi$, it follows that

$$\frac{df}{d\xi} = \sum_{k=0}^{N} h \frac{\partial F_k}{\partial \xi} + \sum_{k=0}^{N} h \lambda_k^T \frac{\partial R_k}{\partial \xi}. \qquad (4.27)$$

This total derivative is numerically verified on a test problem with 12 different performance metrics of interest using the complex-step method (see Figure 4.3).

## 4.3 Backward Difference Formulas (BDF)

The BDF method was first proposed by Curtiss [14] and Henrici [15]. The BDF method is an implicit multistep method based on finite differences: the higher-order difference operators

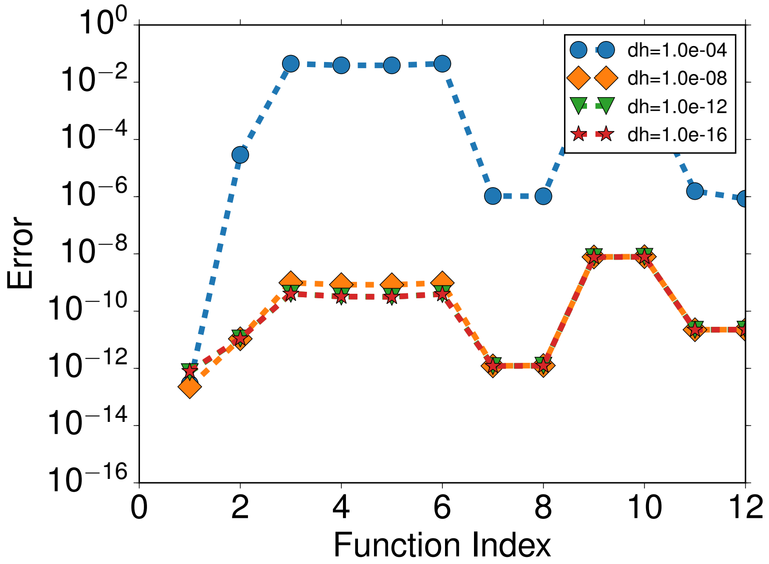


*Figure 4.3: Complex-step verification of the Newmark adjoint scheme for 12 different functions of interest with various perturbation step sizes.*

are obtained by repetitive application of the first-order difference operator. As a result, the first-derivative approximation requires state variable values at $p+1$ points, while the second derivative approximation requires state variables at $2p+1$ points, where $p$ is the order of accuracy. The BDF method is suitable for the solution of stiff ODEs and DAEs and several solution packages, such as `ODEPACK/LSODE` [9], and `DASSL` [10], employ this method. For a BDF method that uses constant step size $h$ the interpolation weights are shown in Table 4.3.

### 4.3.1 Solution of the State Variables

*Table 4.3: BDF interpolation weights up to an approximation order of six.*

| Order $p$ | $\alpha_{p0}$ | $\alpha_{p1}$ | $\alpha_{p2}$ | $\alpha_{p3}$ | $\alpha_{p4}$ | $\alpha_{p5}$ | $\alpha_{p6}$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | -1 | | | | | |
| 2 | 3/2 | -2 | 1/2 | | | | |
| 3 | 11/6 | -3 | 3/2 | -1/3 | | | |
| 4 | 25/12 | -4 | 3 | -4/3 | 1/4 | | |
| 5 | 137/60 | -5 | 5 | -10/3 | 5/4 | -1/5 | |
| 6 | 49/20 | -6 | 15/2 | -20/3 | 15/4 | -6/5 | 1/6 |

#### *4.3.1.1 State Approximation Hypothesis*

The primary unknowns of the BDF time marching scheme are the state variables $q_k$ at each time step $k$. The first and second time derivatives of the state variables $\dot{q}_k$ and $\ddot{q}_k$ are obtained using

$$\begin{aligned} \dot{q}_k &= \sum_{i=0}^{p} \frac{\alpha_i}{h} q_{k-i} + \mathcal{O}(h^p), \\ \ddot{q}_k &= \sum_{i=0}^{2p} \frac{\beta_i}{h^2} q_{k-i} + \mathcal{O}(h^p). \end{aligned} \tag{4.28}$$

The coefficients $\alpha_i$ and $\beta_i$ depend on the order of approximation, $p$. The first and second time derivatives of the state variables are linear combinations of the state variables scaled with BDF coefficients, as illustrated in Figure 4.4.

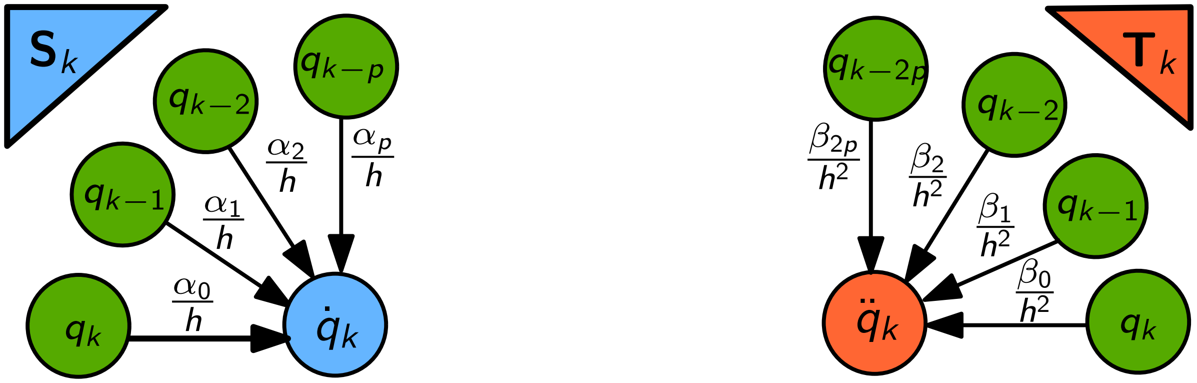


*Figure 4.4: A weighted linear combination of the state variables with scaled BDF coefficients yielding the first (left) and second time derivatives of state variables (right). These relations are respectively labeled as the state approximation equations $S_k$ and $T_k$.*

#### *4.3.1.2 Solution of the Nonlinear System*

Once the time derivatives of states have been approximated using Equation (4.28), the implicit system of nonlinear equations at $k-$th time step becomes

$$R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k) = 0. \tag{4.29}$$

This nonlinear system of equations is solved iteratively using Newton's method. The iterative updates to the unknown variables are obtained by solving a linearization of the governing equations with respect to the primary unknowns $q_k$ as follows

$$\left[\frac{\beta_0}{h^2}\frac{\partial R_k}{\partial \ddot{q}_k} + \frac{\alpha_0}{h}\frac{\partial R_k}{\partial \dot{q}_k} + \frac{\partial R_k}{\partial q_k}\right]\Delta q_k = -R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k). \tag{4.30}$$

The secondary and tertiary updates to the first and second time derivatives of the state variables are readily obtained by scaling the state variable update $\Delta q_k$ using the BDF coefficients, at each iteration $n$ of the nonlinear solution. The resulting update formulas to the state variables are

$$\begin{aligned} q_k^{n+1} &= q_k^n + \Delta q_k^n, \\ \dot{q}_k^{n+1} &= \dot{q}_k^n + \frac{\alpha_0}{h}\Delta q_k^n, \\ \ddot{q}_k^{n+1} &= \ddot{q}_k^n + \frac{\beta_0}{h^2}\Delta q_k^n. \end{aligned} \tag{4.31}$$

The use of the secondary and tertiary updates in Equation (4.31) is preferred since the original backwards difference formulas (4.28) typically require more vector operations. The iterative updates to the state variables and their first and second time derivatives are continued until the governing equations are solved to a specified tolerance. The accuracy of adjoint derivatives rely on the accuracy of the solution of the governing equations. Therefore it is important that the discrete nonlinear system (4.29) is solved to a tight tolerance.

### 4.3.2 Solution of the Adjoint Variables

#### *4.3.2.1 Formation of the Lagrangian*

The adjoint equations are derived using a Lagrangian formulation. The time integral of the functional of interest is discretized as follows,

$$f(\xi) = \int_0^T F\left(t, \xi, \ddot{q}, \dot{q}, q\right)\ \mathrm{d}t \approx \sum_{k=0}^{N} hF_k\left(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k\right). \tag{4.32}$$

The inner product of the governing equations with the adjoint variables, $\lambda$, is approximated as follows

$$\int_0^T \lambda^T R(t, \xi, \ddot{q}, \dot{q}, q)\ \mathrm{d}t \approx \sum_{k=0}^{N} h \lambda_k^T R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k). \tag{4.33}$$

The state approximation residuals of the BDF method are introduced as follows

$$\begin{aligned} S_k &= \sum_{i=0}^{p} \frac{\alpha_i}{h} q_{k-i} - \dot{q}_k, \\ T_k &= \sum_{i=0}^{2p} \frac{\beta_i}{h^2} q_{k-i} - \ddot{q}_k. \end{aligned} \tag{4.34}$$

The adjoint variables $\psi_k$ and $\phi_k$ are associated with state approximation residuals $S_k$ and $T_k$, respectively. With these definitions, the Lagrangian is defined as the following linear combination:

$$\mathcal{L} = \sum_{k=0}^{N} h F_k + \sum_{k=0}^{N} h \lambda_k^T R_k + \sum_{k=0}^{N} \psi_k^T S_k + \sum_{k=0}^{N} \phi_k^T T_k. \tag{4.35}$$

The adjoint variables $\lambda_k$, $\psi_k$ and $\phi_k$, are the unknown weights in the linear combination (4.35). Once they are determined the total derivative is readily available as a linear combination involving the same weights. The formation of Lagrangian is illustrated in Figure 4.5 along with corresponding inputs to the equations $R_k$, $S_k$, $T_k$ and $F_k$. The similarity in inputs to the governing equations $R_k$, and the functional, $F_k$ are due to their identical mathematical forms.

#### *4.3.2.2 The Adjoint Equations:*

The system of equations to solve for the adjoint variables is obtained from the stationary points of the Lagrangian with respect to the state variables and their first and second time derivatives at each time step. The number of terms in the adjoint system of equations as a function of the order of BDF method is listed in Table 4.4.

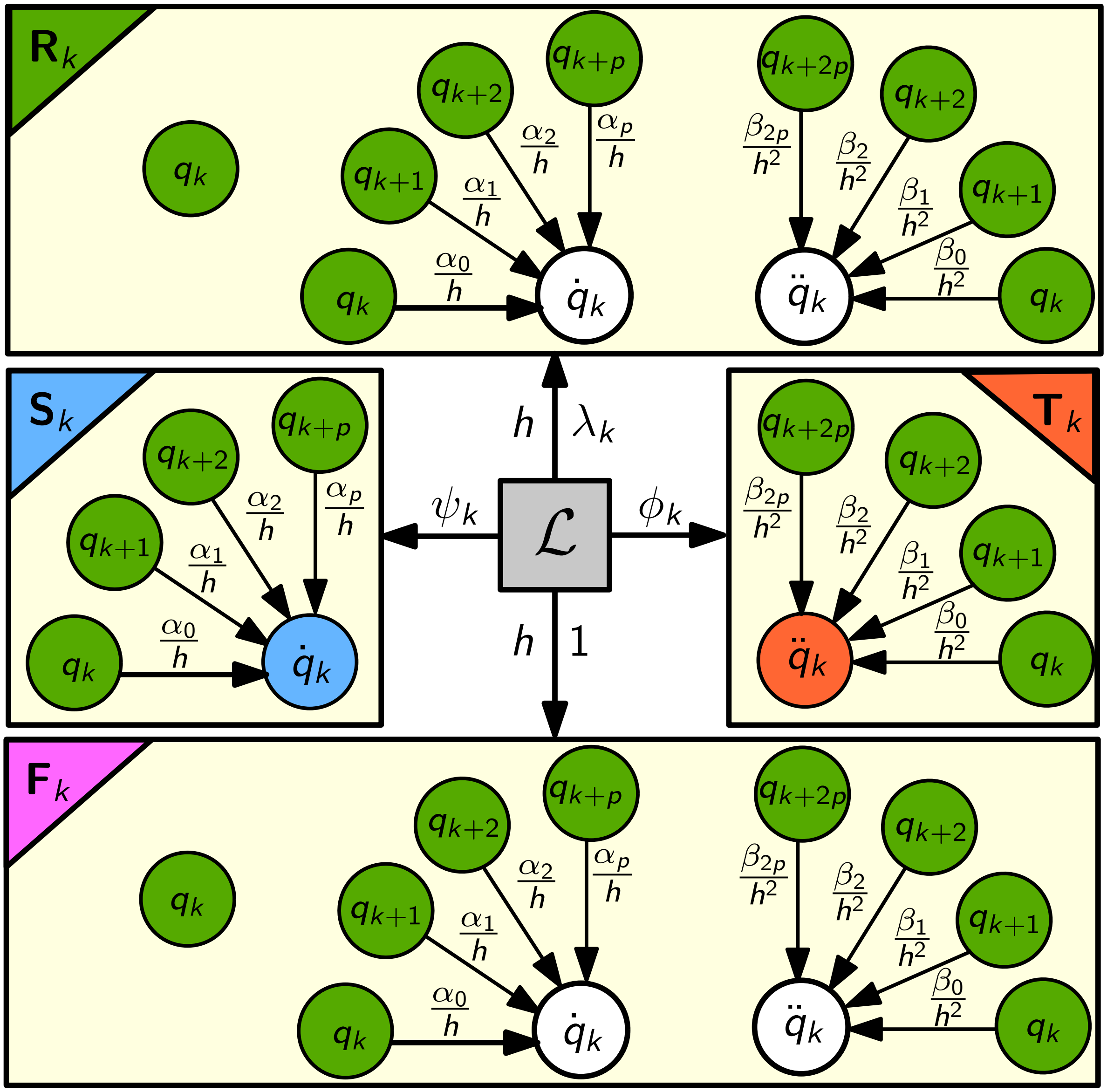


*Figure 4.5: A graphical illustration of the weighted linear combination of equations with corresponding adjoint variables forming the Lagrangian for the BDF method.*

**1. Equation for $\phi_k$:** The equations to solve for $\phi_k$ are obtained using $\partial\mathcal{L}/\partial\ddot{q}_k = 0$. It follows that $\frac{\partial T_k}{\partial \ddot{q}_k}^T \phi_k = 0$, which simplifies further to $\phi_k = 0$.

**2. Equation for $\psi_k$:** The equations to solve for $\psi_k$ are obtained using $\partial\mathcal{L}/\partial\dot{q}_k = 0$. It follows that $\frac{\partial S_k}{\partial \dot{q}_k}^T \psi_k = 0$, which simplifies to $\psi_k = 0$.

*Table 4.4: The number of terms in the adjoint system of equations for p-th order BDF method.*

| | $T$ | $S$ | $R$ | F | Total |
|---|---|---|---|---|---|
| $\phi_k$ | 1 | | | | 1 |
| $\psi_k$ | | 1 | | | 1 |
| $\lambda_k$ | $2p+1$ | $p+1$ | $3p+3$ | $3p+3$ | $9p+8$ |

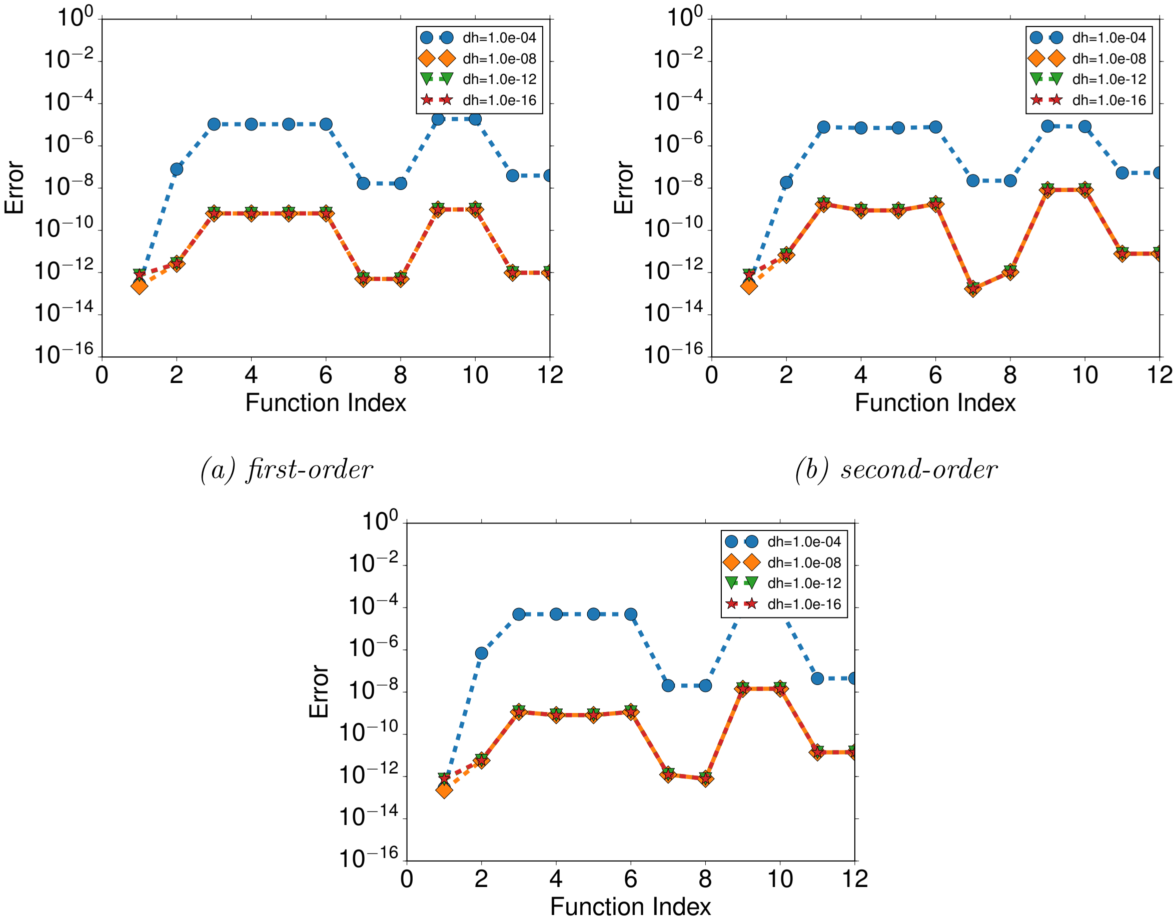
**3. Equation for $\lambda_k$:** The set of equations to solve for $\lambda_k$ are obtained using $\partial\mathcal{L}/\partial q_k = 0$. It follows that

$$\begin{aligned} 0 = \; & h\left[\frac{\beta_0}{h^2}\frac{\partial R_k}{\partial \ddot{q}_k} + \frac{\alpha_0}{h}\frac{\partial R_k}{\partial \dot{q}_k} + \frac{\partial R_k}{\partial q_k}\right]^T \lambda_k + h\sum_{i=1}^{p}\frac{\alpha_i}{h}\frac{\partial R_{k+i}}{\partial \dot{q}_{k+i}}^T \lambda_{k+i} + h\sum_{i=1}^{2p}\frac{\beta_i}{h^2}\frac{\partial R_{k+i}}{\partial \ddot{q}_{k+i}}^T \lambda_{k+i} \\ & + h\left\{\frac{\beta_0}{h^2}\frac{\partial F_k}{\partial \ddot{q}_k} + \frac{\alpha_0}{h}\frac{\partial F_k}{\partial \dot{q}_k} + \frac{\partial F_k}{\partial q_k}\right\}^T + h\sum_{i=1}^{p}\frac{\alpha_i}{h}\frac{\partial F_{k+i}}{\partial \dot{q}_{k+i}}^T + h\sum_{i=1}^{2p}\frac{\beta_i}{h^2}\frac{\partial F_{k+i}}{\partial \ddot{q}_{k+i}}^T \\ & + \sum_{i=0}^{p}\frac{\alpha_i}{h}\psi_{k+i} + \sum_{i=0}^{2p}\frac{\beta_i}{h^2}\phi_{k+i} \end{aligned} \tag{4.36}$$

The $9p+8$ terms in Equation (4.36) arise from the occurrences of the states, $q_k$, as arguments to equations at different time steps, as illustrated in Figure 4.5. The contributions due to $\psi_k$ and $\phi_k$ are zero, which eliminates $3p+2$ terms. Finally, rearranging the terms and dividing by $h$ yields the following linear system to solve for $\lambda_k$:

$$\begin{aligned} \left[\frac{\beta_0}{h^2}\frac{\partial R_k}{\partial \ddot{q}_k} + \frac{\alpha_0}{h}\frac{\partial R_k}{\partial \dot{q}_k} + \frac{\partial R_k}{\partial q_k}\right]^T \lambda_k = & -\left\{\frac{\beta_0}{h^2}\frac{\partial F_k}{\partial \ddot{q}_k} + \frac{\alpha_0}{h}\frac{\partial F_k}{\partial \dot{q}_k} + \frac{\partial F_k}{\partial q_k}\right\}^T \\ & - \sum_{i=1}^{p}\frac{\alpha_i}{h}\frac{\partial R_{k+i}}{\partial \dot{q}_{k+i}}^T \lambda_{k+i} - \sum_{i=1}^{2p}\frac{\beta_i}{h^2}\frac{\partial R_{k+i}}{\partial \ddot{q}_{k+i}}^T \lambda_{k+i} \\ & - \sum_{i=1}^{p}\frac{\alpha_i}{h}\frac{\partial F_{k+i}}{\partial \dot{q}_{k+i}}^T - \sum_{i=1}^{2p}\frac{\beta_i}{h^2}\frac{\partial F_{k+i}}{\partial \ddot{q}_{k+i}}^T, \end{aligned} \tag{4.37}$$

with $6p+6$ terms.



*(a) first-order*

*(b) second-order*

*(c) third-order*

*Figure 4.6: Complex-step verification of the BDF adjoint scheme for 12 different functions of interest with various perturbation step sizes.*

#### *4.3.2.3 Evaluation of Total Derivative:*

Once the adjoint variables have been determined, the total derivative of the functional of interest with respect to the design variables, $\xi$, is the following linear combination:

$$\frac{df}{d\xi} = \sum_{k=0}^{N} h\frac{\partial F_k}{\partial \xi} + \sum_{k=0}^{N} h\lambda_k^T \frac{\partial R_k}{\partial \xi} + \sum_{k=0}^{N} \psi_k^T \frac{\partial S_k}{\partial \xi} + \sum_{k=0}^{N} \phi_k^T \frac{\partial T_k}{\partial \xi}. \tag{4.38}$$

Since, the state approximation equations $S_k$ and $T_k$ are independent of the design variables, $\xi$, it follows that

$$\frac{df}{d\xi} = \sum_{k=0}^{N} h \frac{\partial F_k}{\partial \xi} + \sum_{k=0}^{N} h \lambda_k^T \frac{\partial R_k}{\partial \xi}. \tag{4.39}$$

This total derivative is numerically verified on a test problem using complex-step method (see Figure 4.6).

## 4.4 Adams–Bashforth–Moulton

The Adams–Bashforth–Moulton (see Bashforth and Adams [21] and Moulton [22]) family of linear multistep methods use the past solution values to construct the solution at current step. The ABM methods are based on numerical integration of the polynomial that interpolates solution values. The number of values used to construct the solution determines the order of accuracy and stability of the method. The ABM method is a part of packages such as `EPISODE` [8] and `LSODE` [9], and has been applied to solve stiff problems. The interpolation coefficients of the implicit ABM method for constant step size $h$ are shown in Table 4.5.

*Table 4.5: Implicit ABM coefficients upto an approximation order of six.*

| Order $p$ | $\alpha_{p0}$ | $\alpha_{p1}$ | $\alpha_{p2}$ | $\alpha_{p3}$ | $\alpha_{p4}$ | $\alpha_{p5}$ |
|---|---|---|---|---|---|---|
| 1 | 1 | | | | | |
| 2 | 1/2 | 1/2 | | | | |
| 3 | 5/12 | 8/12 | -1/12 | | | |
| 4 | 9/24 | 19/24 | -5/24 | 1/24 | | |
| 5 | 251/720 | 646/720 | -264/720 | 106/720 | -19/720 | |
| 6 | 475/1440 | 1427/1440 | -798/1440 | 482/1440 | -173/1440 | 27/1440 |

### 4.4.1 Solution of the State Variables

#### *4.4.1.1 State Approximation Hypothesis*

The primary unknowns of the ABM family of methods are the acceleration state variables $\ddot{q}_k$ at each time step $k$. The first time derivative of state variables, $\dot{q}_k$, are obtained by numerical integration of the second time derivative of states as follows

$$\dot{q}_k = \dot{q}_{k-1} + \sum_{i=0}^{p-1} h\alpha_i \ddot{q}_{k-i} + \mathcal{O}(h^p). \tag{4.40}$$

The state variables, $q_k$, are obtained by numerical integration of the first time derivative of states as follows

$$q_k = q_{k-1} + \sum_{i=0}^{p-1} h\alpha_i \dot{q}_{k-i} + \mathcal{O}(h^p). \tag{4.41}$$

The schematic representation of the ABM state approximations is shown in Figure 4.7.

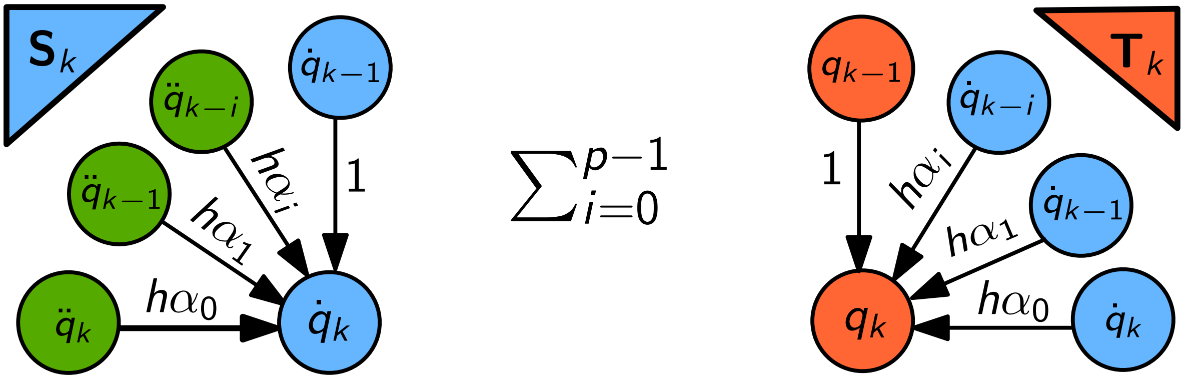


*Figure 4.7: A weighted linear combination of state variables with scaled ABM coefficients yielding the first time derivative of states (left) and the state variables (right).*

### *4.4.1.2 Solution of the Nonlinear System:*

The system of governing equations $R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k) = 0$ is linearized with respect to the primary unknown variable $\ddot{q}_k$ of the ABM time marching scheme at each time step $k$ as follows:

$$\left[\frac{\partial R_k}{\partial \ddot{q}_k} + h\alpha_0 \frac{\partial R_k}{\partial \dot{q}_k} + h^2\alpha_0^2 \frac{\partial R_k}{\partial q_k}\right] \Delta\ddot{q}_k = -R_k(t_k, \xi, \ddot{q}_k, \dot{q}_k, q_k). \tag{4.42}$$

The state variables and the first time derivatives are approximated using Equations (4.41) and (4.40), for an estimated value of $\ddot{q}_k$. The linearization of the governing equations (4.42) is then solved for the primary update $\Delta\ddot{q}_k^n$, at each iteration, $n$, of the nonlinear solution.

The secondary updates to state variables are readily obtained by scaling the primary update using the ABM coefficients. The resulting update formulas to the state variables and their time derivatives are

$$\begin{aligned} \ddot{q}_k^{n+1} &= \ddot{q}_k^n + \Delta\ddot{q}_k^n, \\ \dot{q}_k^{n+1} &= \dot{q}_k^n + h\alpha_0\Delta\ddot{q}_k^n, \\ q_k^{n+1} &= q_k^n + h^2\alpha_0^2\Delta\ddot{q}_k^n. \end{aligned} \tag{4.43}$$

The iterative updates to the state variables and their time derivatives are continued until the governing equations are solved to required tolerance.

### 4.4.2 Solution of the Adjoint Variables

#### *4.4.2.1 Formation of the Lagrangian*

The governing equations and functional of interest follow same treatment discussed previously for other methods. The state approximations of the ABM time marching scheme given in equations (4.40) and (4.41) are expressed as the following residuals

$$\begin{aligned} S_k &= \dot{q}_{k-1} + h\sum_{i=0}^{p-1}\alpha_i\ddot{q}_{k-i} - \dot{q}_k, \\ T_k &= q_{k-1} + h\alpha_0\left(\dot{q}_{k-1} + \sum_{i=0}^{p-1}h\alpha_i\ddot{q}_{k-i}\right) + \sum_{i=1}^{p-1}h\alpha_i\dot{q}_{k-i} - q_k. \end{aligned} \tag{4.44}$$

The term $h\alpha_0\dot{q}_k$ is expanded out in terms of the primary unknown $\ddot{q}_k$ for eliminating the coupling of adjoint equations within each time step. The adjoint variables $\lambda_k$, $\psi_k$ and $\phi_k$ are introduced as respective unknown weights, to the governing equations, $R_k$, the state approximation equations, $S_k$, and $T_k$, arising from the ABM scheme, for each time step, $k$. The geometric intuition in forming the Lagrangian is as follows. All four functions $F_k$, $R_k$, $S_k$ and $T_k$ lie within the span of $\ddot{q}(t,\xi) \otimes \dot{q}(t,\xi) \otimes q(t,\xi) \otimes \xi \otimes t$. The goal is to represent the function of interest $F_k$ as a linear combination of other functions $R_k$, $S_k$ and $T_k$ from the same subspace by associating weights $\lambda_k$, $\psi_k$ and $\phi_k$. The Lagrangian is written as

$$\mathcal{L} = \sum_{k=0}^{N} hF_k + \sum_{k=0}^{N} h\lambda_k^T R_k + \sum_{k=0}^{N} \psi_k^T S_k + \sum_{k=0}^{N} \phi_k^T T_k. \tag{4.45}$$

The formation of the Lagrangian for the ABM method is illustrated in Figure 4.8. The number of terms that occur in the adjoint system of equations for each adjoint variable (see Table 4.6) can be determined graphically from Figure 4.8.

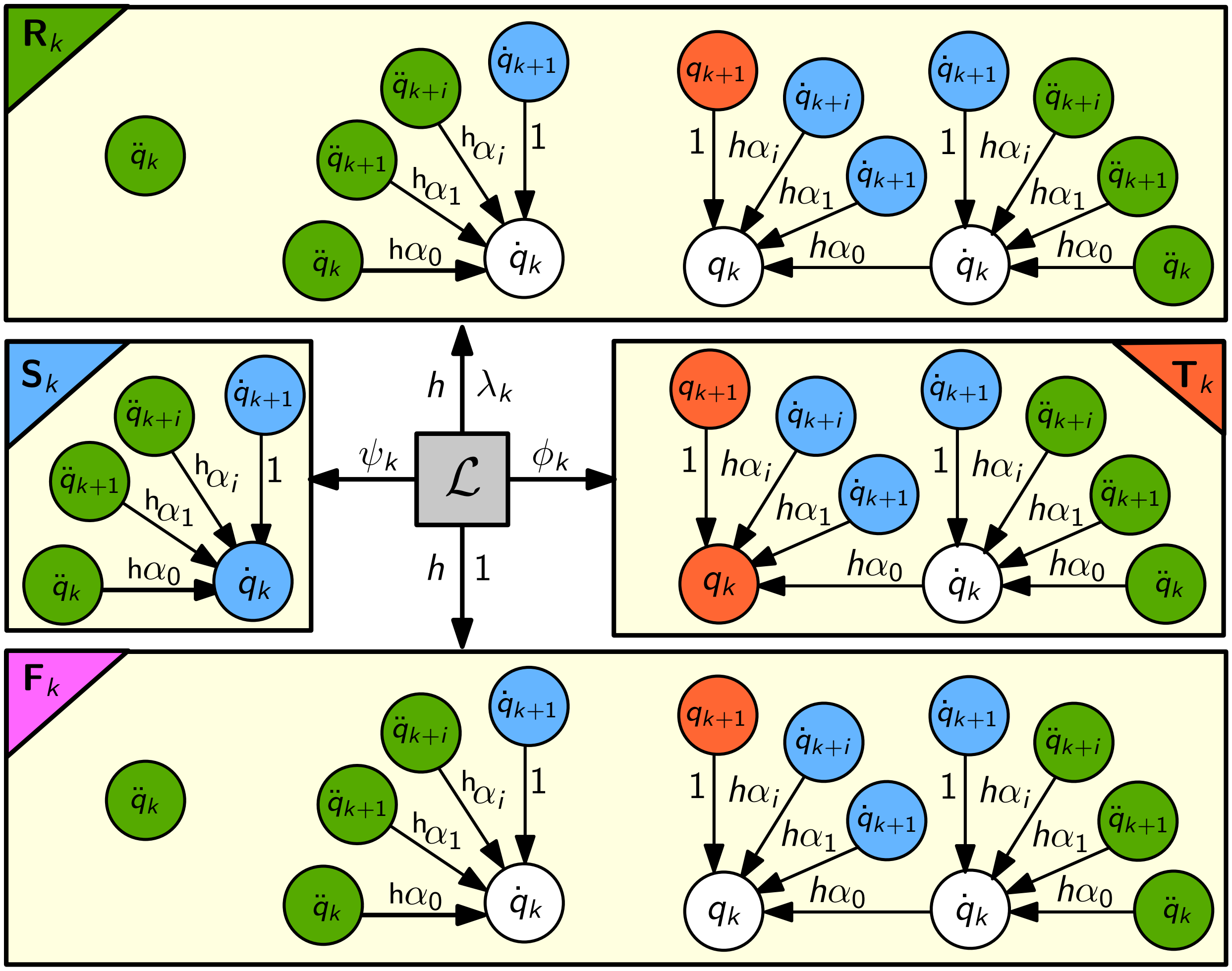


*Figure 4.8: A weighted linear combination of equations with corresponding adjoint variables forming the Lagrangian for ABM method.*

*Table 4.6: Table listing the number of terms in the adjoint system of equations for ABM method.*

| | $T$ | $S$ | $R$ | F | Total |
|---|---|---|---|---|---|
| $\phi_k$ | 2 | | 1 | 1 | 4 |
| $\psi_k$ | p | 2 | p+1 | p+1 | 3p+4 |
| $\lambda_k$ | p | p | 2p+1 | 2p+1 | 6p+2 |

#### *4.4.2.2 The Adjoint Equations:*

The system of equations to solve for the adjoint variables is obtained from the stationary points of the Lagrangian with respect to the position, velocity and acceleration state variables, at each time step, $k$.

**1. Equation for $\phi_k$:** Setting $\partial\mathcal{L}/\partial q_k = 0$ yields

$$\frac{\partial T_k}{\partial q_k}^T \phi_k + \frac{\partial T_{k+1}}{\partial q_k}^T \phi_{k+1} + h\frac{\partial R_{k+1}}{\partial q_k}^T \lambda_{k+1} + h\frac{\partial F_{k+1}}{\partial q_k}^T = 0. \tag{4.46}$$

This simplifies to

$$\phi_k = \phi_{k+1} + h\left[\frac{\partial R_{k+1}}{\partial q_{k+1}}\right]^T \lambda_{k+1} + h\left\{\frac{\partial F_{k+1}}{\partial q_{k+1}}\right\}^T. \tag{4.47}$$

**2. Equation for $\psi_k$:** Setting $\partial\mathcal{L}/\partial\dot{q}_k = 0$ yields

$$\frac{\partial S_k}{\partial \dot{q}_k}^T \psi_k + \frac{\partial S_{k+1}}{\partial \dot{q}_k}^T \psi_{k+1} + \sum_{i=1}^{p-1}\frac{\partial T_{k+i}}{\partial \dot{q}_k}^T \phi_{k+i} + h\sum_{i=1}^{p-1}\frac{\partial R_{k+i}}{\partial \dot{q}_k}^T \lambda_{k+i} + h\sum_{i=1}^{p-1}\frac{\partial F_{k+i}}{\partial \dot{q}_k}^T = 0. \tag{4.48}$$

Further simplifications result in the adjoint variable, $\psi_k$, as a linear combination:

$$\begin{aligned}\psi_k = \psi_{k+1} + h\alpha_0\phi_{k+1} + h\left[\frac{\partial R_{k+1}}{\partial \dot{q}_{k+1}} + h\alpha_0\frac{\partial R_{k+1}}{\partial q_{k+1}}\right]^T \lambda_{k+1} + h\left\{\frac{\partial F_{k+1}}{\partial \dot{q}_{k+1}} + h\alpha_0\frac{\partial F_{k+1}}{\partial q_{k+1}}\right\}^T \\ + h\sum_{i=1}^{p-1}\alpha_i\phi_{k+i} + h\sum_{i=1}^{p-1}\left[h\alpha_i\frac{\partial R_{k+i}}{\partial q_{k+i}}\right]^T \lambda_{k+i} + h\sum_{i=1}^{p-1}\left\{h\alpha_i\frac{\partial F_{k+i}}{\partial q_{k+i}}\right\}^T.\end{aligned} \tag{4.49}$$

**3. Equation for $\lambda_k$:** Setting $\partial \mathcal{L}/\partial \ddot{q}_k = 0$ yields the equation to solve for the adjoint variable, $\lambda_k$,

$$h \sum_{i=0}^{p-1} \frac{\partial R_{k+i}}{\partial \ddot{q}_k}^T \lambda_{k+i} + h \sum_{i=0}^{p-1} \frac{\partial F_{k+i}}{\partial \ddot{q}_k}^T + \sum_{i=0}^{p-1} \frac{\partial S_{k+i}}{\partial \ddot{q}_k}^T \psi_{k+i} + \sum_{i=0}^{p-1} \frac{\partial T_{k+i}}{\partial \ddot{q}_k}^T \phi_{k+i} = 0. \tag{4.50}$$

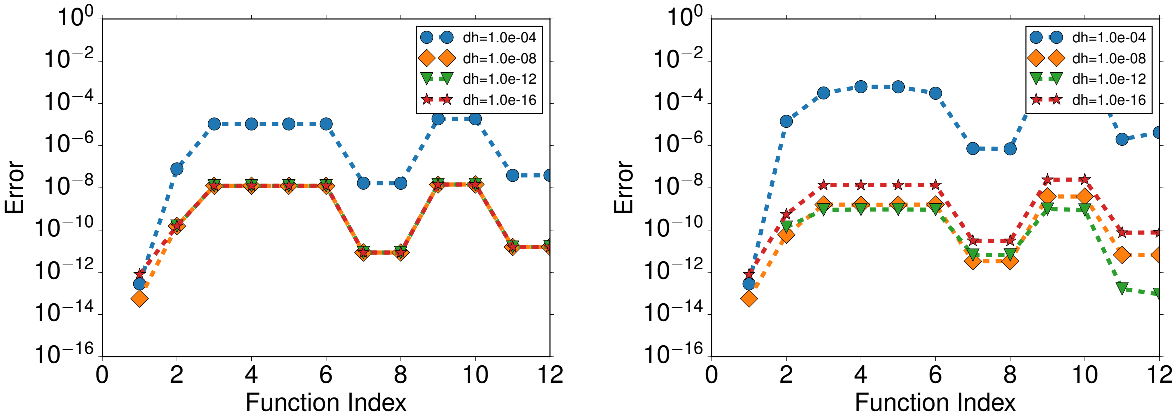
Expanding the derivative terms, separating out current and previous determined terms, dividing by $h$ and rearranging for the unknown adjoint variable $\lambda_k$ results in the following linear system:

$$\begin{aligned}\left[\frac{\partial R_k}{\partial \ddot{q}_k} + h\alpha_0 \frac{\partial R_k}{\partial \dot{q}_k} + h^2\alpha_0^2 \frac{\partial R_k}{\partial q_k}\right]^T \lambda_k = &- \left\{\frac{\partial F_k}{\partial \ddot{q}_k} + h\alpha_0 \frac{\partial F_k}{\partial \dot{q}_k} + h^2\alpha_0^2 \frac{\partial F_k}{\partial q_k}\right\}^T \\ &- \frac{1}{h}\left\{h\alpha_0 \psi_k + h^2\alpha_0^2 \phi_k\right\} \\ &- \sum_{i=1}^{p-1} \left[h\alpha_i \frac{\partial R_{k+i}}{\partial \dot{q}_{k+i}} + h\alpha_0 h\alpha_i \frac{\partial R_{k+i}}{\partial q_{k+i}}\right]^T \lambda_{k+i} \\ &- \sum_{i=1}^{p-1} \left\{h\alpha_i \frac{\partial F_{k+i}}{\partial \dot{q}_{k+i}} + h\alpha_0 h\alpha_i \frac{\partial F_{k+i}}{\partial q_{k+i}}\right\}^T \\ &- \frac{1}{h}\sum_{i=1}^{p-1} \left\{h\alpha_i \psi_{k+i} + h\alpha_0 h\alpha_i \phi_{k+i}\right\}.\end{aligned} \tag{4.51}$$

The partial derivative terms in the linear system correspond to the occurrences of $\ddot{q}_k$ as inputs to equations $R$, $S$, $T$ and $F$ in the time history, as illustrated in Figure 4.8. The determination of adjoint variables allows evaluating the total derivative of the functional of interest with respect to the design variables, using Equation (4.39). See Figure 4.9 for numerical verification of the total derivative using complex-step method on a simple test problem.

## 4.5 Diagonally Implicit Runge–Kutta

Runge–Kutta methods belong to the class of multistage methods for solving differential equations. They are termed as multistage methods as they require solutions at intermediate



*(a) first-order*

*(b) second-order*

*Figure 4.9: Complex-step verification of the ABM adjoint scheme for 12 different functions of interest with various perturbation step sizes.*

stages to progress from step $t_{k-1}$ to $t_k$. Butcher [23] extended explicit Runge–Kutta method to Implicit Runge–Kutta (IRK) method, and furthermore Alexander [24] and Cash [25] developed Diagonally Implicit Runge–Kutta (DIRK) method. The coefficients that define the DIRK method are arranged in a tabular format as shown in Table 4.7, and is commonly referred to as the Butcher's tableau. The lower triangular nature of the tableau enables the successive solution of the nonlinear governing equations at each stage. Note that the tableau fully populated in the case of an IRK scheme, resulting in full coupling among all stages. Due to the availability of *one-stage-second-order*, *two-stage-third-order* and *three-stage-fourth-order* DIRK methods developed by Alexander [24], it is preferred to IRK methods.

*Table 4.7: Butcher's tableau of DIRK coefficients.*

| Stage | $\beta_1$ | $\beta_2$ | $\cdots$ | $\beta_s$ | |
|---|---|---|---|---|---|
| 1 | $\alpha_{11}$ | 0 | 0 | 0 | $\tau_1$ |
| 2 | $\alpha_{21}$ | $\alpha_{22}$ | 0 | 0 | $\tau_2$ |
| | $\vdots$ | $\vdots$ | $\ddots$ | 0 | $\vdots$ |
| s | $\alpha_{s1}$ | $\alpha_{s2}$ | $\cdots$ | $\alpha_{ss}$ | $\tau_s$ |

### 4.5.1 Solution of the State Variables

The development of DIRK scheme for second-order descriptor systems and the corresponding time dependent discrete-adjoint are discussed next.

#### *4.5.1.1 Stage Approximations Hypothesis*

The governing equations are solved at intermediate time steps, $t_{ki}$, referred to as the *stages*. The intermediate stage state variables and their first and second time derivatives are denoted as $u_{ki}$, $\dot{u}_{ki}$ and $\ddot{u}_{ki}$, respectively. The stage state approximation relations are

$$\begin{aligned} \dot{u}_{ki} &= \dot{q}_{k-1} + h\sum_{j=1}^{i}\alpha_{ij}\ddot{u}_{kj}, \\ u_{ki} &= q_{k-1} + h\sum_{j=1}^{i}\alpha_{ij}\dot{u}_{kj}. \end{aligned} \tag{4.52}$$

The indices $i$ and $j$ to refer to row and column of the coefficients in the Butcher's tableau shown in Table 4.7. The schematic representation of the intermediate stage approximations is shown in Figure 4.10.

$\ddot{u}_{ki}$
$\dot{q}_{k-1}$
$h\alpha_{ii}$
1
$\ddot{u}_{k2}$
$h\alpha_{i2}$
$\ddot{u}_{k1}$
$h\alpha_{i1}$
$\dot{u}_{ki}$
$q_{k-1}$
1
$h\alpha_{ii}$
$\dot{u}_{ki}$
$\dot{u}_{k2}$
$h\alpha_{i2}$
$h\alpha_{i1}$
$u_{ki}$
$\dot{u}_{k1}$

*Figure 4.10: The intermediate stage state variables of DIRK are formed as a linear combination.*

#### *4.5.1.2 State Approximation Hypothesis*

The state variables at time, $t_k$, are expressed as follows

$$\begin{aligned}\ddot{q}_k &= \sum_{i=1}^{s} \beta_i \ddot{u}_{ki},\\ \dot{q}_k &= \dot{q}_{k-1} + h\sum_{i=1}^{s} \beta_i \ddot{u}_{ki},\\ q_k &= q_{k-1} + h\sum_{i=1}^{s} \beta_i \dot{u}_{ki}\end{aligned} \tag{4.53}$$

and is illustrated in Figure 4.11.

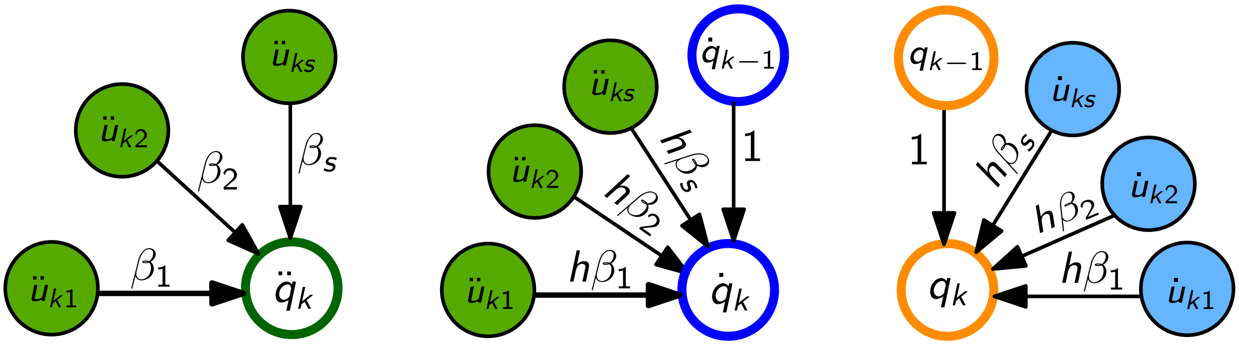


*Figure 4.11: The state variables and their time derivatives at k-th time step formed as a linear combination.*

#### *4.5.1.3 Solution of the Nonlinear System*

The nonlinear system of equations to be solved, at each stage, $i$, and time step, $k$, is $R_{ki}(t_{ki}, \xi, \ddot{u}_{ki}, \dot{u}_{ki}, u_{ki}) = 0$. This can be solved in a manner similar to Newmark and ABM methods discussed previously. The key difference here is that the nonlinear system is solved at $s$-intermediate stages for each time step. The linearized form of the nonlinear system to be solved repeatedly at each stage is,

$$\left[\frac{\partial R_{ki}}{\partial \ddot{u}_{ki}} + h\alpha_{ii}\frac{\partial R_{ki}}{\partial \dot{u}_{ki}} + h^2\alpha_{ii}^2\frac{\partial R_{ki}}{\partial u_{ki}}\right]\Delta\ddot{u}_{ki} = -R_{ki}(t_{ki}, \xi, \ddot{u}_{ki}, \dot{u}_{ki}, u_{ki}). \tag{4.54}$$

The update formulas for the stage state variables are

$$\begin{aligned}\ddot{u}_{ki}^{n+1} &= \ddot{u}_{ki}^{n} + \Delta\ddot{u}_{ki}^{n},\\ \dot{u}_{ki}^{n+1} &= \dot{u}_{ki}^{n} + h\alpha_{ii}\Delta\ddot{u}_{ki}^{n},\\ u_{ki}^{n+1} &= u_{ki}^{n} + h^2\alpha_{ii}^2\Delta\ddot{u}_{ki}^{n}.\end{aligned} \tag{4.55}$$

The iterative updates to the stage state variables and derivatives continue until the governing equations are solved to the required tolerance.

### 4.5.2 Solution of the Adjoint Variables

#### *4.5.2.1 Formation of the Lagrangian:*

The state approximations of DIRK time marching scheme given in Equation (4.53) are reformulated as the following residuals

$$\begin{aligned}S_k &= \dot{q}_{k-1} + h\sum_{i=1}^{s}\beta_i\ddot{u}_{ki} - \dot{q}_k,\\ T_k &= q_{k-1} + h\sum_{i=1}^{s}\beta_i\dot{u}_{ki} - q_k.\end{aligned} \tag{4.56}$$

The adjoint variables $\lambda_{ki}$, $\psi_k$ and $\phi_k$ are associated with the governing equations at each stage, $R_{ki}$, and the state approximation equations, $S_k$, and $T_k$, arising from the DIRK scheme, for each time step, $k$. The Lagrangian is written as

$$\mathcal{L} = \sum_{k=0}^{N} h\sum_{i=1}^{s}\beta_i F_{ki} + \sum_{k=0}^{N} h\sum_{i=1}^{s}\beta_i\lambda_{ki}^T R_{ki} + \sum_{k=0}^{N}\psi_k^T S_k + \sum_{k=0}^{N}\phi_k^T T_k. \tag{4.57}$$

The governing equations and functions of interest follow slightly different treatment for DIRK since it is a multistage method. Note that $h$ arises from the discretization of the continuous integral into $N$ time intervals and $\beta_i$ arises from the discretization of each time interval into $s$ stages. The formation of the Lagrangian is schematically shown in Figure 4.12. The

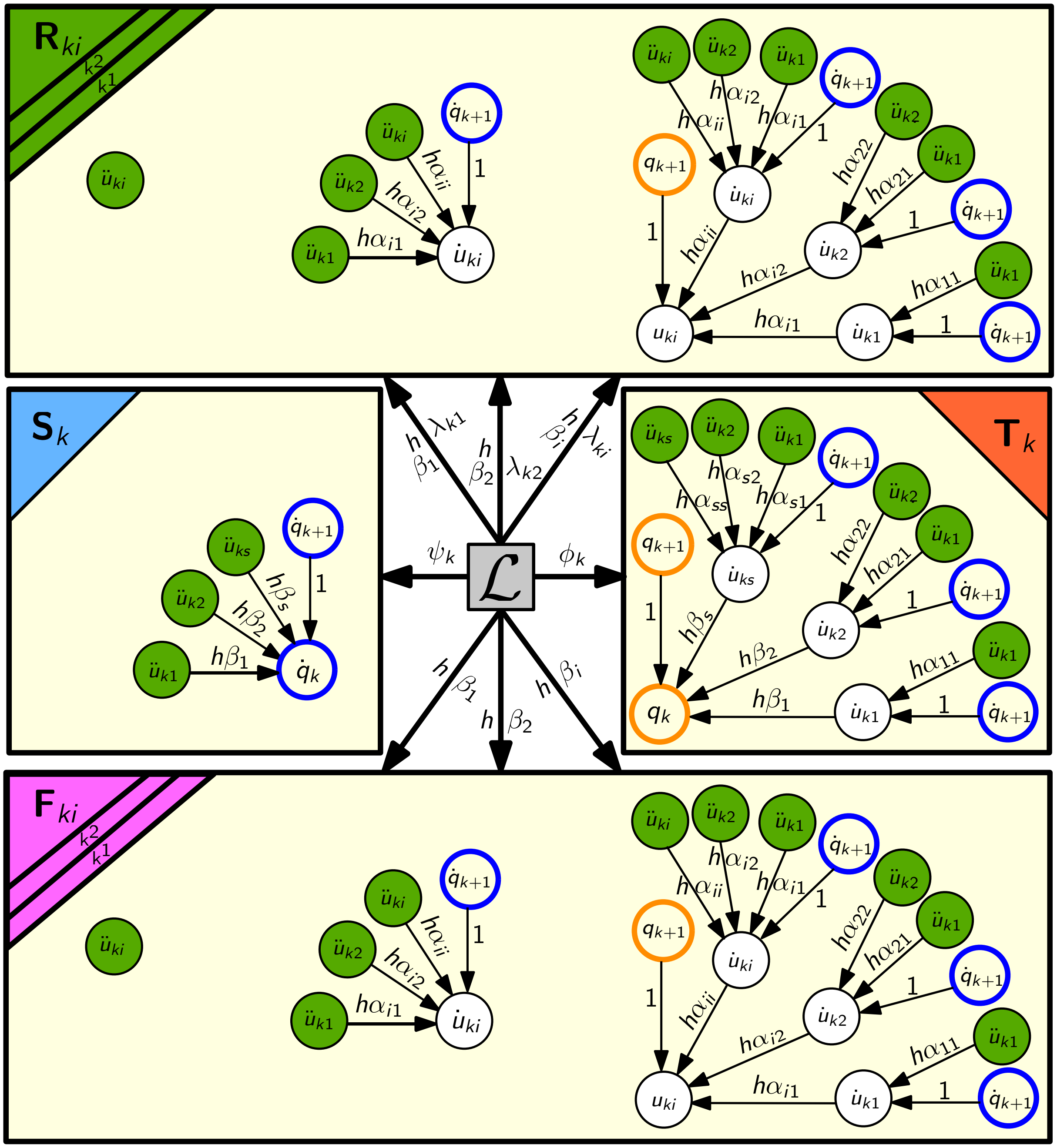


*Figure 4.12: A weighted linear combination of equations with corresponding adjoint variables forming the Lagrangian for DIRK method.*

number of terms that occur in the adjoint system of equations for each adjoint variable can be determined graphically from Figure 4.12 and is listed in Table 4.8. The number of terms listed in Table 4.8 can be seen to exist in the following adjoint system of equations.

*Table 4.8: Table listing the number of terms in the adjoint system of equations for DIRK method.*

| | $T$ | $S$ | $R$ | F | Total |
|---|---|---|---|---|---|
| $\phi_k$ | 2 | | s | s | 2+2s |
| $\psi_k$ | s | 2 | 2s | 2s | 2+5s |
| $\lambda_k$ | s-i+1 | 1 | 2(s-i)+1 | 2(s-i)+1 | 5(s-i)+1 |

**1. Equation for $\phi_k$:** Setting $\partial \mathcal{L}/\partial q_k = 0$ yields

$$\frac{\partial T_k}{\partial q_k}^T \phi_k + \frac{\partial T_{k+1}}{\partial q_k}^T \phi_{k+1} + h\sum_{i=1}^{s} \beta_i \frac{\partial R_{k+1,i}}{\partial q_k}^T \lambda_{k+1,i} + h\sum_{i=1}^{s} \beta_i \frac{\partial F_{k+1,i}}{\partial q_k}^T = 0. \tag{4.58}$$

This simplifies to

$$\phi_k = \phi_{k+1} + h\sum_{i=1}^{s} \beta_i \frac{\partial R_{k+1,i}}{\partial u_{k+1,i}}^T \lambda_{k+1,i} + h\sum_{i=1}^{s} \beta_i \frac{\partial F_{k+1,i}}{\partial u_{k+1,i}}^T . \tag{4.59}$$

**2. Equation for $\psi_k$:** Setting $\partial \mathcal{L}/\partial \dot{q}_k = 0$ yields

$$\frac{\partial S_k}{\partial \dot{q}_k}^T \psi_k + \frac{\partial S_{k+1}}{\partial \dot{q}_k}^T \psi_{k+1} + \sum_{i=1}^{s} \beta_i \frac{\partial T_{k+1}}{\partial \dot{q}_k}^T \phi_{k+1} + h\sum_{i=1}^{s} \beta_i \frac{\partial R_{k+1,i}}{\partial \dot{q}_k}^T \lambda_{k+1,i} + h\sum_{i=1}^{s} \beta_i \frac{\partial F_{k+1,i}}{\partial \dot{q}_k}^T = 0. \tag{4.60}$$

This becomes

$$\begin{aligned}\psi_k = \psi_{k+1} + h\sum_{i=1}^{s} \beta_i \phi_{k+1} + h\sum_{i=1}^{s} \beta_i \left[ \frac{\partial R_{k+1,i}}{\partial \dot{u}_{k+1,i}} + h\sum_{j=1}^{i} \alpha_{ij} \frac{\partial R_{k+1,i}}{\partial u_{k+1,i}} \right]^T \lambda_{k+1,i} \\ + h\sum_{i=1}^{s} \beta_i \left\{ \frac{\partial F_{k+1,i}}{\partial \dot{u}_{k+1,i}} + h\sum_{j=1}^{i} \alpha_{ij} \frac{\partial F_{k+1,i}}{\partial u_{k+1,i}} \right\}^T .\end{aligned} \tag{4.61}$$

Using the properties of DIRK coefficients: $\sum_{i=1}^{s} \beta_i = 1$ and $\sum_{j=1}^{i} \alpha_{ij} = \tau_i$,

$$\begin{aligned}\psi_k = \psi_{k+1} + h\phi_{k+1} + h\sum_{i=1}^{s} \beta_i \left[ \frac{\partial R_{k+1,i}}{\partial \dot{u}_{k+1,i}} + h\tau_i \frac{\partial R_{k+1,i}}{\partial u_{k+1,i}} \right]^T \lambda_{k+1,i} \\ + h\sum_{i=1}^{s} \beta_i \left\{ \frac{\partial F_{k+1,i}}{\partial \dot{u}_{k+1,i}} + h\tau_i \frac{\partial F_{k+1,i}}{\partial u_{k+1,i}} \right\}^T .\end{aligned} \tag{4.62}$$

**3. Equation for $\lambda_{ki}$:** Taking $\partial\mathcal{L}/\partial\ddot{u}_{ki} = 0$ yields

$$h\sum_{j=i}^{s}\beta_j\frac{\partial R_{kj}}{\partial \ddot{u}_{ki}}^T\lambda_{kj} + h\sum_{j=i}^{s}\beta_j\frac{\partial F_{kj}}{\partial \ddot{u}_{ki}}^T + \frac{\partial S_k}{\partial \ddot{u}_{ki}}^T\psi_k + \frac{\partial T_k}{\partial \ddot{u}_{ki}}^T\phi_k = 0. \tag{4.63}$$

Applying chain rule and expanding the terms results in

$$\begin{aligned}0 = {}& h\beta_i\frac{\partial R_{ki}}{\partial \ddot{u}_{ki}} + h\sum_{j=i+1}^{s}\beta_j\left[h\alpha_{ji}\frac{\partial R_{kj}}{\partial \dot{u}_{kj}} + h^2\sum_{p=i}^{j}\alpha_{jp}\alpha_{pi}\frac{\partial R_{kj}}{\partial u_{kj}}\right]^T\lambda_{kj}\\ & + h\beta_i\frac{\partial F_{ki}}{\partial \ddot{u}_{ki}} + h\sum_{j=i+1}^{s}\beta_j\left\{h\alpha_{ji}\frac{\partial F_{kj}}{\partial \dot{u}_{kj}} + h^2\sum_{p=i}^{j}\alpha_{jp}\alpha_{pi}\frac{\partial F_{kj}}{\partial u_{kj}}\right\}^T\\ & + h\beta_i\psi_k \quad + \left(h\sum_{j=i}^{s}\beta_j h\alpha_{ji}\right)\phi_k.\end{aligned} \tag{4.64}$$

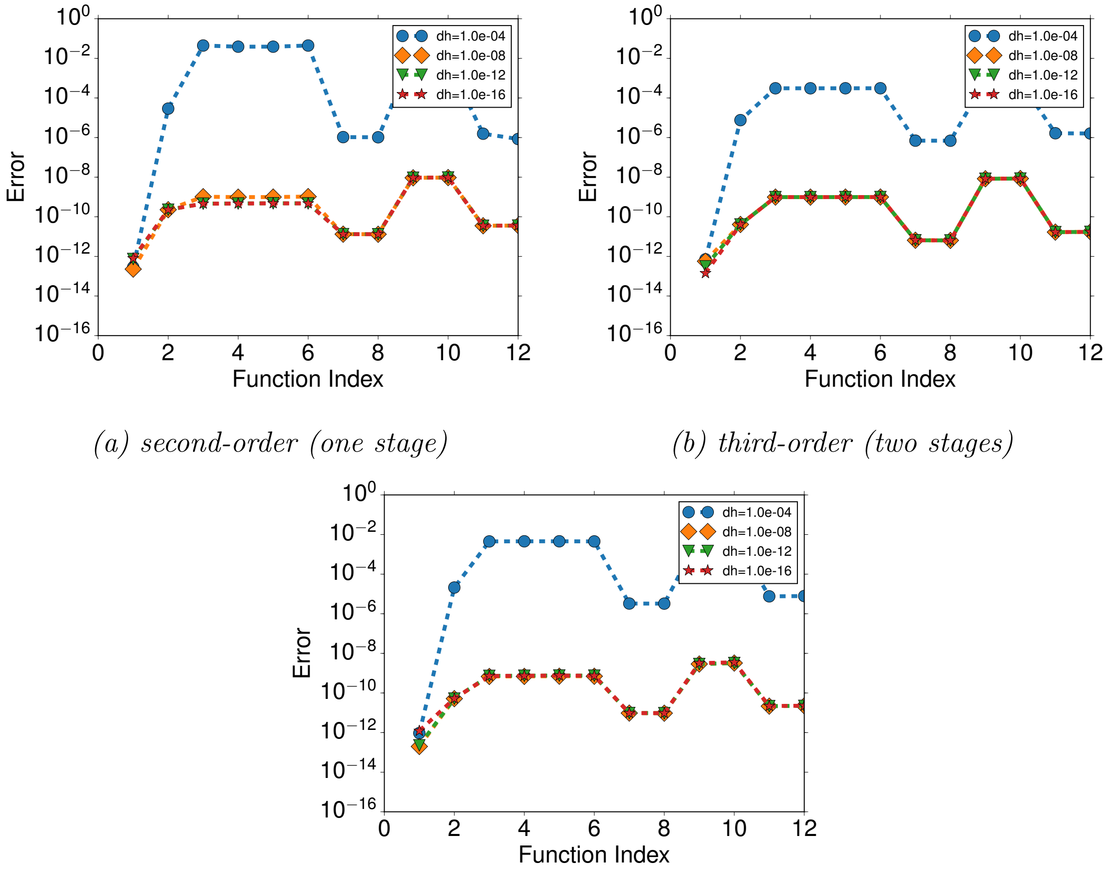
Dividing by $h$ and rearranging for the unknown adjoint variable $\lambda_{ki}$ results in the following linear system

$$\begin{aligned}\beta_i\left[\frac{\partial R_{ki}}{\partial \ddot{u}_{ki}} + h\alpha_{ii}\frac{\partial R_{ki}}{\partial \dot{u}_{ki}} + h^2\alpha_{ii}^2\frac{\partial R_{ki}}{\partial u_{ki}}\right]^T\lambda_{ki} = {}& -\beta_i\left[\frac{\partial F_{ki}}{\partial \ddot{u}_{ki}} + h^2\alpha_{ii}^2\frac{\partial F_{ki}}{\partial \dot{u}_{ki}} + h^2\alpha_{ii}^2\frac{\partial F_{ki}}{\partial u_{ki}}\right]^T\\ & - \sum_{j=i+1}^{s}\beta_j\left[h\alpha_{ji}\frac{\partial R_{kj}}{\partial \dot{u}_{kj}} + h^2\sum_{p=i}^{j}\alpha_{jp}\alpha_{pi}\frac{\partial R_{kj}}{\partial u_{kj}}\right]^T\lambda_{kj}\\ & - \sum_{j=i+1}^{s}\beta_j\left\{h\alpha_{ji}\frac{\partial F_{kj}}{\partial \dot{u}_{kj}} + h^2\sum_{p=i}^{j}\alpha_{jp}\alpha_{pi}\frac{\partial F_{kj}}{\partial u_{kj}}\right\}^T\\ & - \beta_i\psi_k - \left(\sum_{j=i}^{s}\beta_j h\alpha_{ji}\right)\phi_k.\end{aligned} \tag{4.65}$$

The total derivative is computed in an analogous manner to other methods as follows

$$\frac{dF}{d\xi} = \sum_{k=0}^{N} h\sum_{i=1}^{s}\beta_i\frac{\partial F_{ki}}{\partial \xi} + \sum_{k=0}^{N} h\sum_{i=1}^{s}\beta_i\lambda_{ki}^T\frac{\partial R_{ki}}{\partial \xi}. \tag{4.66}$$

The numerical verification of total derivative computed using adjoint method (4.66) using complex-step method is shown in Figure 4.13.



*(a) second-order (one stage)*

*(b) third-order (two stages)*

*(c) fourth-order (three stages)*

*Figure 4.13: Complex-step verification of the DIRK adjoint scheme for 12 different functions of interest with various perturbation step sizes.*

## 4.6 Implementation of the Adjoint Method

General flexible multibody dynamics simulation tools contain a large library of flexible and rigid elements, joints, dampers, and a wide variety of kinematic constraints that can be used to model multibody systems. The implementation of the discrete adjoint imposes additional requirements on each component of the simulation. These additional requirements must be handled carefully in order to maintain an efficient and accurate adjoint implementation. This section presents the organization and implementation of the proposed adjoint sensitivities, that is designed to be modular and extensible to facilitate an expanding library of flexible and rigid elements in TACS [3]. The adjoint equations derived in this chapter contain:

1. The derivatives of the *function* of interest and the *governing equations* with respect to the *state variables*,

2. The derivatives of the *function* of interest and the *governing equations* with respect to the *design variables*, and

3. The products of the *adjoint variables* with the derivatives the *governing equations* respect to the *state variables.*

These three primary terms are implemented using a library that contains four interfaces: `Element`, `Function`, `Assembler` and `Integrator`. The organization and relationships between these four interfaces are shown in Figure 4.14. This organization allows for the separation of functionality that enables the underlying element and function library to be extended without having to change the adjoint implementation. The functionality of these interfaces are explained in the remainder of this section.

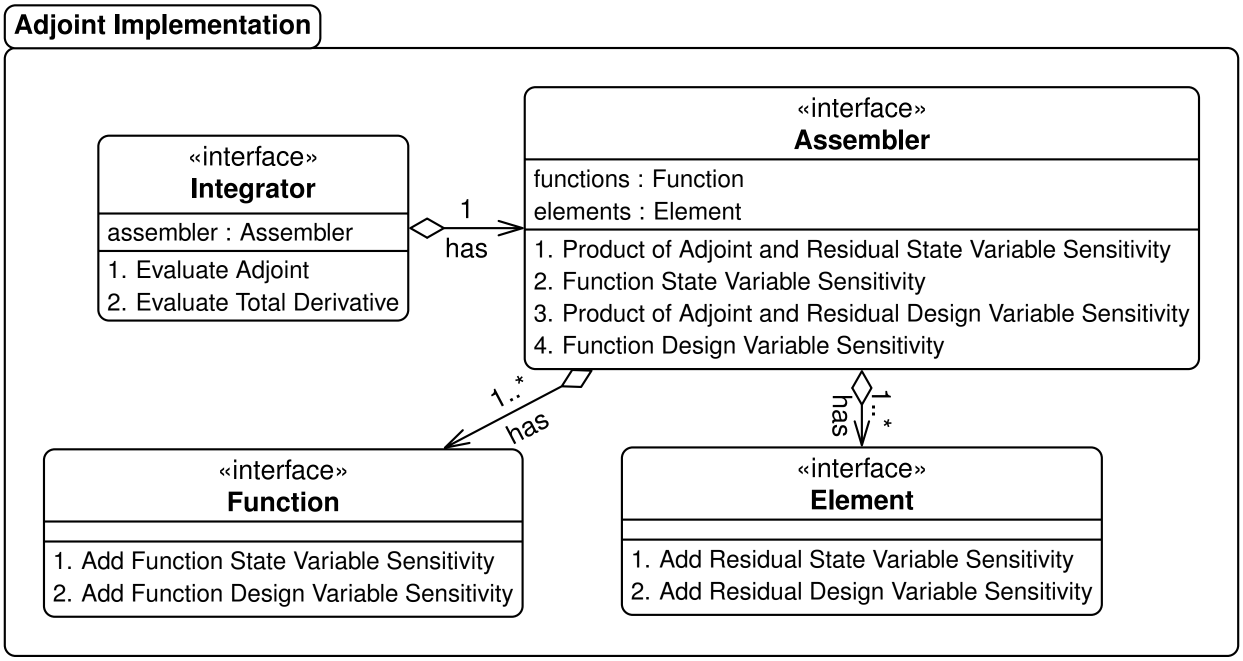

Adjoint Implementation
«interface»
Integrator
assembler : Assembler
1. Evaluate Adjoint
2. Evaluate Total Derivative
1
has
«interface»
Assembler
functions : Function
elements : Element
1. Product of Adjoint and Residual State Variable Sensitivity
2. Function State Variable Sensitivity
3. Product of Adjoint and Residual Design Variable Sensitivity
4. Function Design Variable Sensitivity
1..*
has
1..*
has
«interface»
Function
1. Add Function State Variable Sensitivity
2. Add Function Design Variable Sensitivity
«interface»
Element
1. Add Residual State Variable Sensitivity
2. Add Residual Design Variable Sensitivity

*Figure 4.14: Class diagram illustrating the architecture of adjoint-based gradient implementation.*

### 4.6.1 Element Interface

The element library contains beam, shell, and rigid-body elements as well as kinematic constraints including the lower kinematic pairs. These elements implement a common `Element` interface by providing specific implementations for the abstract prototypes based on the governing equations of motion. This interface contains two routines required for the adjoint implementation:

1. The computation of element-wise Jacobian matrices that are used by the `Assembler` to evaluate the global transpose Jacobian in the linear adjoint system.

2. The evaluation of the derivative of element-wise product of the residuals and the adjoint variables with respect to the design variables. This routine is used to evaluate the total derivative.

As new elements are added to the multibody dynamics library, they are required to implement these two routines so that they can be seamlessly merged in the existing framework.

### 4.6.2 Function Interface

The `Function` interface contains similar prototypes as the `Element` interface. The interface provides the derivatives of function of interests for design optimization which include two primary function-level routines:

1. The evaluation of the element-wise derivative of the functional integrand with respect to the state variables and their time derivatives.

2. The element-wise computation of the derivative of the functional integrand with respect to the design variables, required for the total derivative.

The functions are evaluated over all or a subset of elements in the domain.

### 4.6.3 Assembler Interface

The Assembler interface is designed to operate on a collection of Element and Function instances. The routines provided in this interface assemble the partial derivative terms necessary for sensitivity analysis and place them in global matrices and vectors. These operations depend only on the prototypes defined in Element and Function interfaces, rather than on the specific implementations of element and function types. This dependency of the Assembler on Element and Function interfaces is shown in Figure 4.14.

#### *4.6.3.1 Solving the Adjoint Equations*

The first set of Assembler routines are required for the solution of the adjoint equations. These functions compute the transpose Jacobian-vector product of the governing equations with respect to the state variables

$$\chi \leftarrow \chi + \left[ \gamma \frac{\partial R}{\partial \ddot{q}} + \beta \frac{\partial R}{\partial \dot{q}} + \alpha \frac{\partial R}{\partial q} \right]^T \chi, \tag{4.67}$$

and the derivative of the functional integrand with respect to state variables

$$\chi \leftarrow \chi + \left[ \gamma \frac{\partial F}{\partial \ddot{q}} + \beta \frac{\partial F}{\partial \dot{q}} + \alpha \frac{\partial F}{\partial q} \right]. \tag{4.68}$$

Here $\chi$ is a place-holder for a state vector determined from the context of the adjoint equations. The inputs to these routines are scalar constants for each partial derivative ($\alpha$, $\beta$, and $\gamma$) and the state variables, $q$, and their time derivatives, $\dot{q}$, and $\ddot{q}$. The state variables and their time derivatives are stored to disk during the solution phase and reloaded when marching backwards in time during the adjoint solution process. This reduces the amount of memory required when the number of time steps is large. The routines (4.67) and (4.68) are used frequently in the discrete adjoint implementations.

#### *4.6.3.2 Evaluating the Total Derivative*

The second set of assembly-level routines are needed to evaluate the total derivative of the functional of interest. These routines compute the product of the adjoint variables with the derivative of the governing equations with respect to design variables

$$\chi \leftarrow \chi + \alpha \frac{\partial R}{\partial \xi}^T \chi, \tag{4.69}$$

and the derivative of the functional integrand with respect to design variables

$$\chi \leftarrow \chi + \alpha \frac{\partial F}{\partial \xi}^T . \tag{4.70}$$

Here the inputs consist of a scalar $\alpha$, the design variables $\xi$, and the state variables, $q$, and their time derivatives, $\dot{q}$, and $\ddot{q}$. The output for both of these routines is a vector with the same dimension as the design variable vector. The routines (4.69) and (4.70) are used once at each stage to accumulate the contributions to the total derivative.

### 4.6.4 Integrator Interface

The class implementing the `Integrator` interface completes the evaluation the adjoint variables and the computation of the total derivative and provides it to the optimizer. The `Integrator` interface contains an instance of `Assembler`, which enables it to evaluate the partial derivative terms from the governing equations and the functions of interest, and scale them with the appropriate coefficients, as dictated by the adjoint equations. Note that the `Integrator` does not interact directly with `Element` and `Function` interfaces, but instead uses the `Assembler` interface, as shown in Figure 4.14. This class contains routines that implement DIRK specific operations and is used repeatedly in a time loop starting from the final time step and ending at the initial time step. The `Assembler` set of routines used by the `Integrator` are designed to work for any adjoint method corresponding to other time marching schemes such as Backwards Difference Formula, Adams-Bashforth-Moulton, or Newmark's method. The implementation of other time-integration methods requires only a new implementation of the Integrator interface.

**Summary.** In this chapter, the mathematical details of implicit solution process of initial value problems arising from time dependent systems is discussed. The state approximation hypothesis supplied by the time marching method, is used in conjunction with a generalized Newton–Raphson iteration scheme to solve the implicit nonlinear system. The discrete adjoint equations used to obtain semianalytical sensitivities of functions of interest are presented. In the spirit of generality, the equations are presented for a general order of accuracy $p$ and the abstraction of equations as $R(t, \xi, \ddot{q}, \dot{q}, q)$ and $F(t, \xi, \ddot{q}, \dot{q}, q)$. This allows the application of the derived equations for any time dependent physical system fitting the mathematical form, and solved using multistep/multistage time marching method of $p$-th order of accuracy with a constant step size $h$. Finally, the implementation details of time dependent adjoint equations in a modular manner is presented.

# CHAPTER 5

# MULTIBODY DYNAMICS AND ADJOINT BASED DETERMINISTIC OPTIMIZATION

> For since the fabric of the universe is most perfect and the work of a most wise Creator, nothing at all takes place in the universe in which some rule of maximum or minimum does not appear.
>
> Leonard Euler [1707-1783]

**Introduction.** The intent of this chapter is to show the physical application of the time marching and adjoint sensitivity framework presented in Chapter 4 to design optimization problems in the context of flexible multibody dynamics. We start with simple dynamical system such as a pendulum and build upon complexity by adding flexible bodies, kinematic constraints and motion actuators. We end this chapter with a rotorcraft optimization application.

## 5.1 Triple Pendulum

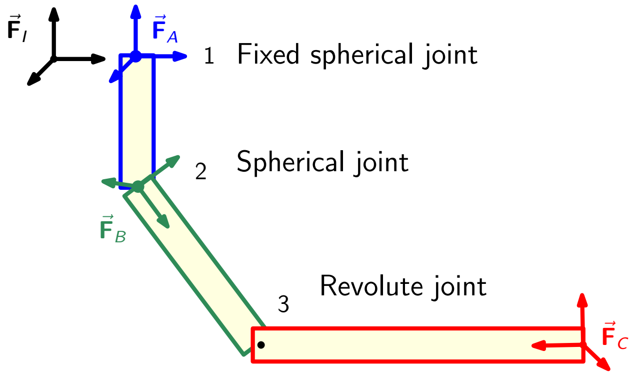


*Figure 5.1: The schematic of the triple pendulum system.*

### 5.1.1 Analysis Setup

Figure 5.1 shows the triple pendulum system with three rigid bodies which are denoted as $\mathcal{B} = \{A, B, C\}$ and three kinematic constraints at the points $\mathcal{P} = \{1, 2, 3\}$. The properties of the bodies and the kinematic constraints are listed in Tables 5.1 and 5.2, respectively. The body axes are chosen such that one of the orthogonal axes are aligned with the geometrical dimensions of the body to simplify the calculation of the inertial properties. The bodies are assumed to have constant material density.

*Table 5.1: List of bodies in the pendulum system and their properties.*

| Body | Type | Mass | Length | Width | Thickness |
|---|---|---|---|---|---|
| A | Rigid | 1.0 | 1.0 | 0.1 | 0.1 |
| B | Rigid | 2.0 | 2.0 | 0.1 | 0.1 |
| C | Rigid | 3.0 | 3.0 | 0.1 | 0.1 |

*Table 5.2: List of kinematic constraints in the pendulum system and their properties.*

| Joint | Type | Components |
|---|---|---|
| 1 | Spherical | $F_I$ and body A |
| 2 | Revolute (hinge) | Bodies A and B |
| 3 | Revolute (hinge) | Bodies B and C |

### 5.1.2 Dynamics

Figure 5.2 shows the timelapse of motion of the bodies in the system over a duration of three seconds. The effect of the revolute joint can be seen where the adjacent bodies in the joint are constrained to rotate about a locally-aligned axis. Figure 5.3 shows the changes in the potential and kinetic energies of the system over a 10 second time interval. Since non-conservative forces are not modeled, such as joint friction, the sum of the potential and kinetic energy should remain constant. Figure 5.3a illustrates the complementary trend of energy transfer between kinetic and potential energies. However, the limited numerical accuracy of the time integration scheme introduces an energy defect that can grow over time.

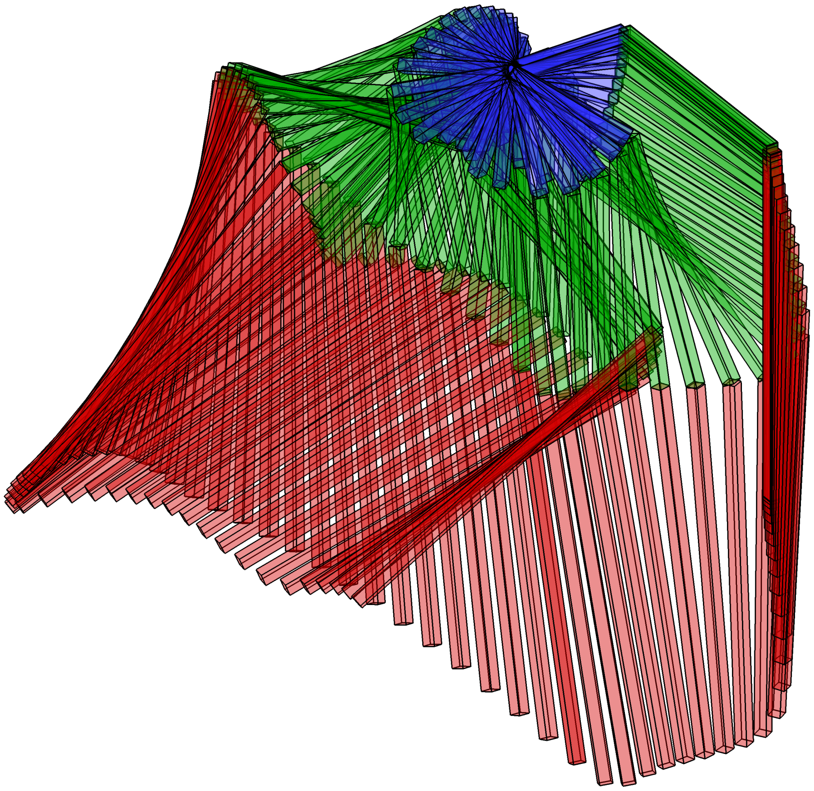
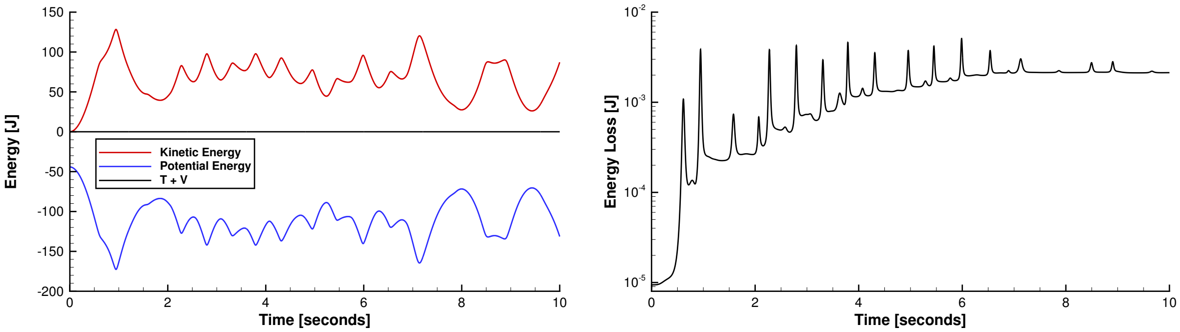


*Figure 5.2: Motion of the triple pendulum over the first 3 seconds.*

To assess this error, Figure 5.3 shows the energy loss over the same time period. Note that over the entire simulation, the energy loss is about $2 \times 10^{-3}$ J.



*(a) Kinetic and potential energy*  *(b) Total energy loss over the time interval*

*Figure 5.3: Plot of the potential, kinetic and total energies with time for the pendulum system.*

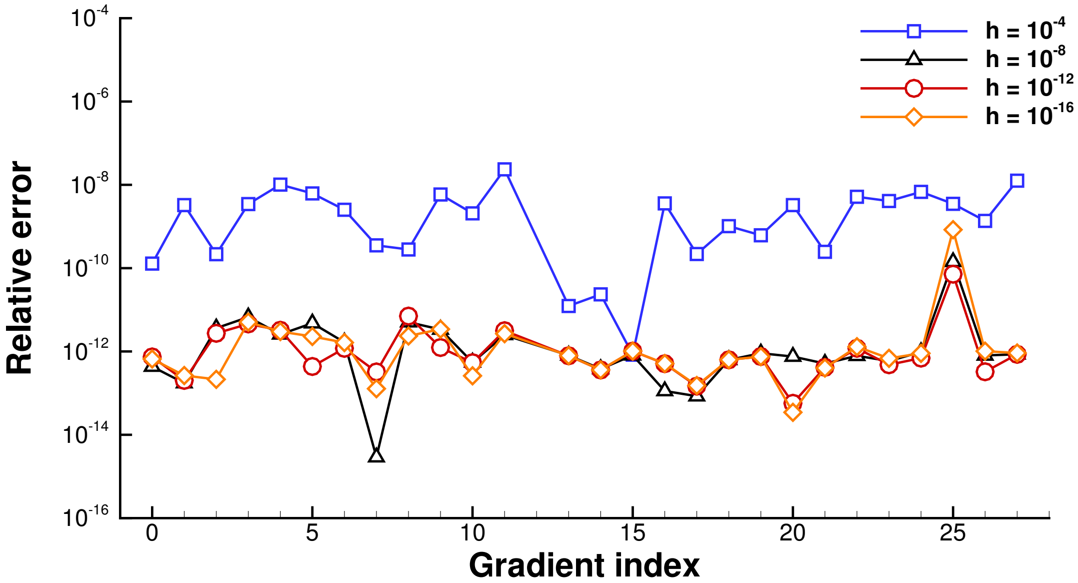


*Figure 5.4: Gradient verification study with the complex-step method using step sizes of* $10^{-4}$*,* $10^{-8}$*,* $10^{-12}$*, and* $10^{-16}$*.*

### 5.1.3 Adjoint Gradients

Figure 5.4 shows a complex-step verification of the adjoint-gradient computed using BDF method. The verification study compares the derivative of Kreisselmeier–Steinhauser [45, 46, 116] aggregation of velocity with respect to a series of design variables consisting of initial configuration variables and inertial properties. The KS function approximates the maximum velocity achieved over the time interval of the simulation. Each component of the gradient exhibits a relative accuracy on the order of $10^{-12}$, illustrating near machine precision accuracy of all gradients.

## 5.2 Trebuchet (Catapult)

Trebuchets have been used for warfare from fifth century B.C till medieval times. In this section we study the multibody dynamics of trebuchet and apply it to an optimization problem of achieving maximum range of the projectile. The trebuchet works to transfer the potential energy of the counterweight to impart kinetic energy to the projectile mass.

### 5.2.1 Analysis Setup

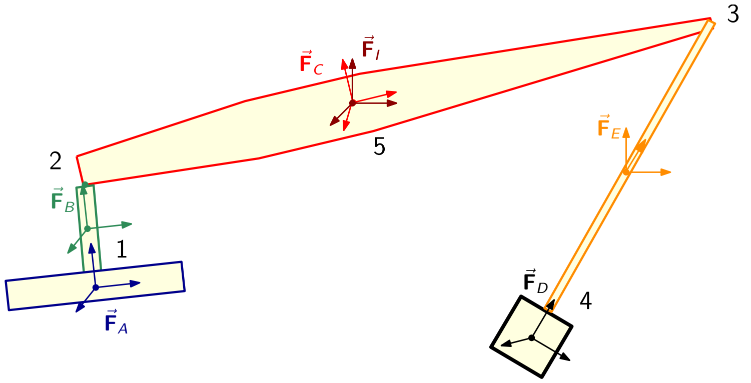


*Figure 5.5: The schematic of the trebuchet system.*

The trebuchet is created using five bodies given by $\mathcal{B} = \{A,\ B,\ C,\ D,\ E\}$, and five kinematic constraints labeled $\mathcal{P} = \{1,\ 2,\ 3,\ 4,\ 5\}$ as shown in Figure 5.5. The kinematic constraints at points 1, 2 and 3 are revolute, while the other joints at points 4 and 5 are modeled as spherical. The whole trebuchet assembly rotates about the axle at point 5. The body axes are chosen to enable convenient calculation of inertial properties. Table 5.3 contains the list of bodies and their geometric/material properties. Figure 5.6 depicts the motion of the trebuchet system obtained using BDF method. The trebuchet arm starts from a horizontal orientation and reaches a near vertical position as it rotates about the axle. Note that the axle is not shown explicitly. The angular momentum of the swinging motion generated by the counterweight is transferred as linear momentum to the projectile mass through the arm and projectile link. Figure 5.7 shows the kinetic and potential energy in the trebuchet

*Table 5.3: List of bodies in the trebuchet system and their properties.*

| Body | Name | Density | Length | Width | Thickness |
|---|---|---|---|---|---|
| A | Counter weight | 25 | 4 | 4 | 1 |
| B | Connecting link | 10 | 0.5 | 0.5 | 2 |
| C | Arm | 2 | 20 | 0.5 | 2 |
| D | Projectile link | $10^{-2}$ | 0.2 | 0.5 | 6 |
| E | Projectile | $10^{-2}$ | 1 | 1 | 1 |

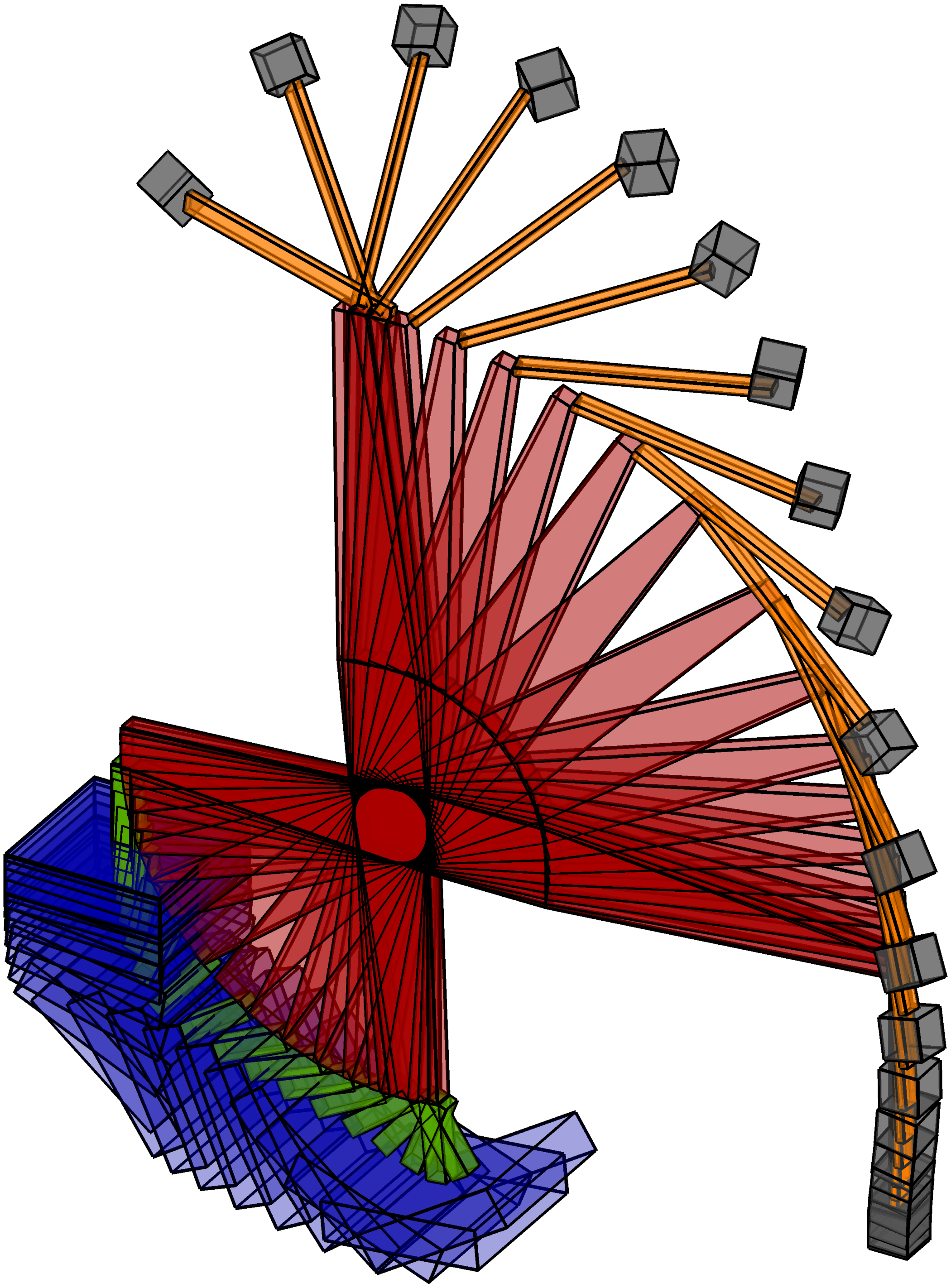
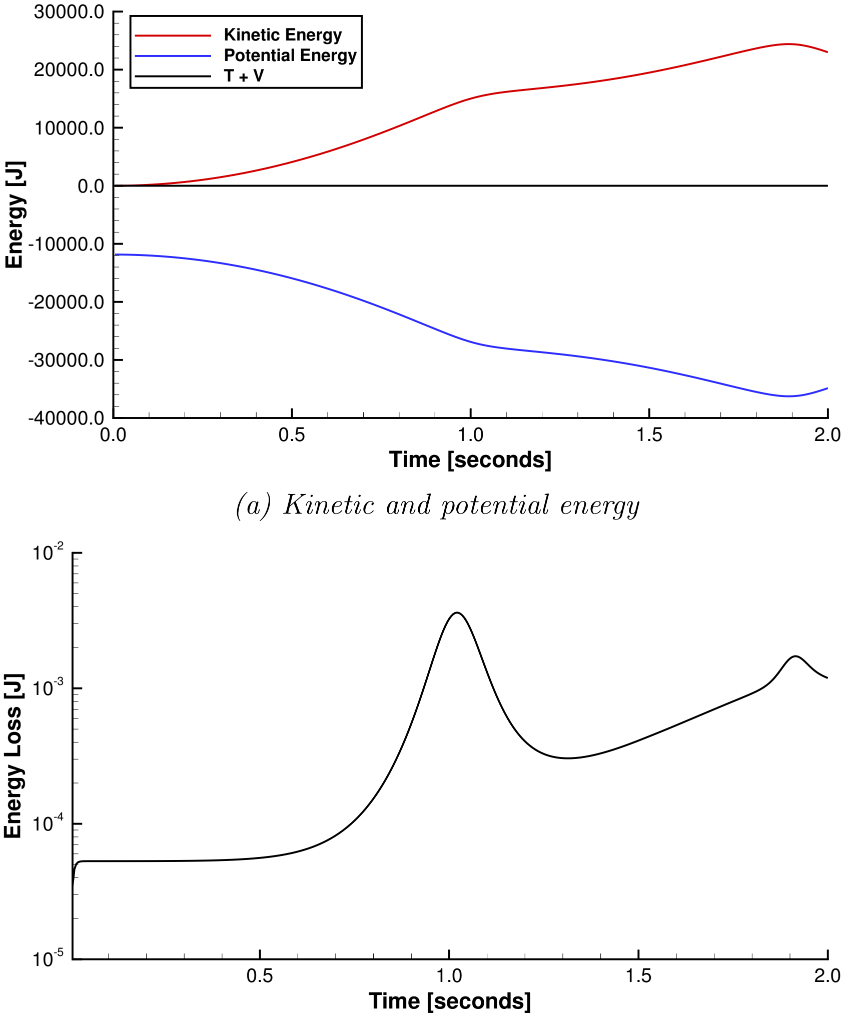


*Figure 5.6: Motion of the trebuchet*

system over the time history of the simulation. During the motion, the potential energy of the system, stored primarily in the counterweight is transferred to kinetic energy. The total energy of the system is conserved, since no non-conservative forces are modeled. The time integration error produces a small change of less than $3 \times 10^{-2}$ J in the total energy in the system, as shown in Figure 5.7. This energy loss can be reduced by utilizing a smaller time integration step size.



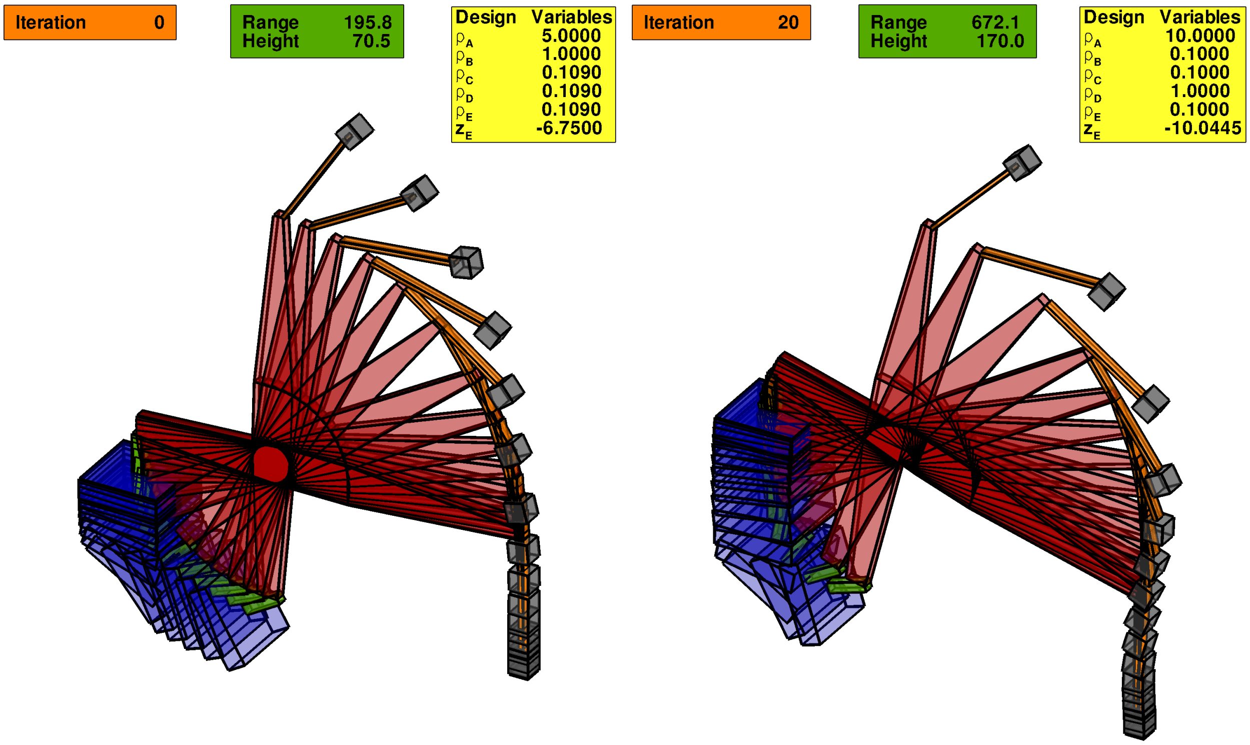
*(a) Kinetic and potential energy*

*(b) Total energy loss over the time interval*

*Figure 5.7: Plot of the potential, kinetic and total energies with time for the trebuchet system.*

### 5.2.2 Trebuchet optimization

The objective of optimization problem is to maximize projectile range, which is estimated using the kinematics of the projectile motion under gravity. The precise release point of the projectile is not calculated. Instead, we use the optimal release point by taking the maximum of the projectile range if it were released at any time during the entire trebuchet motion. We estimate this maximum range using the KS function, in a similar manner to the maximum velocity function described above. We also impose a constraint that the projectile must clear a barrier of specified height at a location down range along the path of the projectile. The present trebuchet problem consists of six design variables and one constraint. The six design variables consist of the mass of the different components within the trebuchet system and an initial condition variable governing the release height of the counterweight. Figure 5.8 shows the initial and optimized trebuchet designs. Note that the counterweight release height is unconstrained at the final design point. The release height is selected such that the motion of the counterweight is synchronized with the arm and projectile motion to achieve maximum velocity at the release point.



*(a) Initial design*

*(b) Optimized design*

*Figure 5.8: Figure illustrating the initial and final trebuchet designs.*

## 5.3 Four-Bar Mechanism

### 5.3.1 Analysis Setup

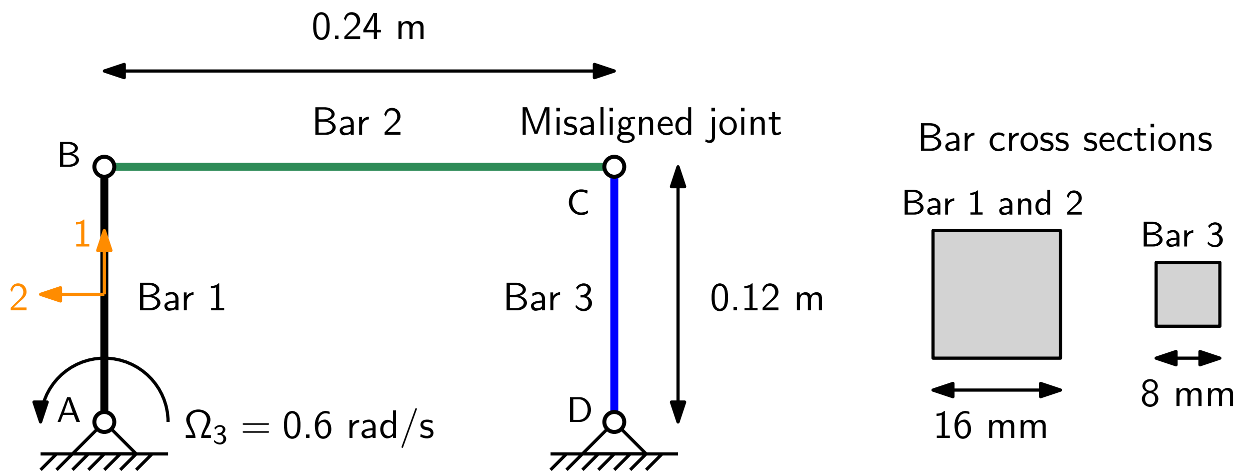


*Figure 5.9: The four-bar mechanism problem used for dynamics verification of TACS.*

Figure 8.14 illustrates the setup of the four-bar mechanism (see Bauchau et al. [1]). Three bars $AB$, $BC$ and $CD$ of the mechanism are joined together using revolute connections. An imaginary fourth bar exists in the mechanism between the points $A$ and $D$. The revolute joints at the points $A$, $B$, and $D$, have an axis of rotation that is perpendicular to the plane of the mechanism. The revolute joint at point $C$ is misaligned by an angle of $5°$, rotated about the direction of the bar $CD$. Bars $AB$ and $BC$ are of the same cross-section, while bar $CD$ has a smaller and more flexible cross section. The bars in the mechanism are modeled using quadratic beam elements that are derived from Timoshenko beam theory. The rotation of bar $AB$ about point $A$ of the mechanism is driven at an angular rate of $\Omega_3 = 0.6\,\text{rad/s}$.

### 5.3.2 Motion and Internal Forces

When the bars are rigid, the four-bar mechanism locks and motion is inhibited. However, when the bars are modeled as flexible, motion becomes possible since the bars can bend to overcome the locking behavior. The motion of the four-bar mechanism is illustrated in Figure 5.10 as a time lapse. If joint C were not misaligned, the bars would rotate in phase.

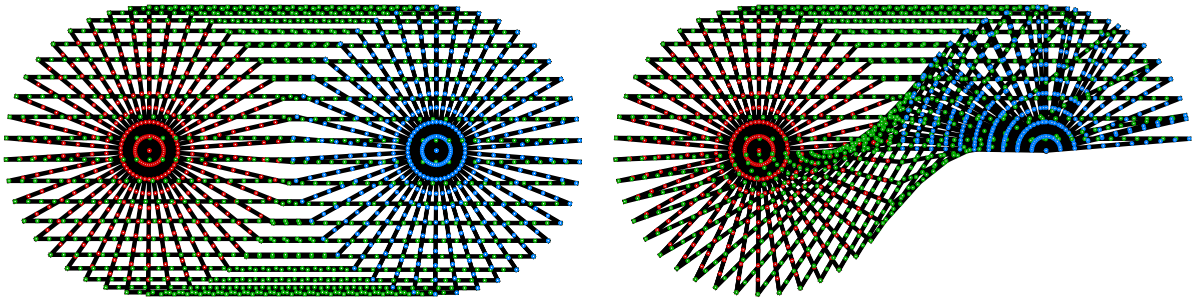


*Figure 5.10: The time evolution of flexible four-bar mechanism*

However, due to the presence of misalignment, the third bar never completes a full rotation, while the first bar drives the motion. Figure 5.11 shows a comparison of the axial force and bending moment in bar $AB$ at the mid-span compared with the same predictions using Dymore [1]. The force and moment predictions can be seen to be in excellent agreement for this benchmark problem.

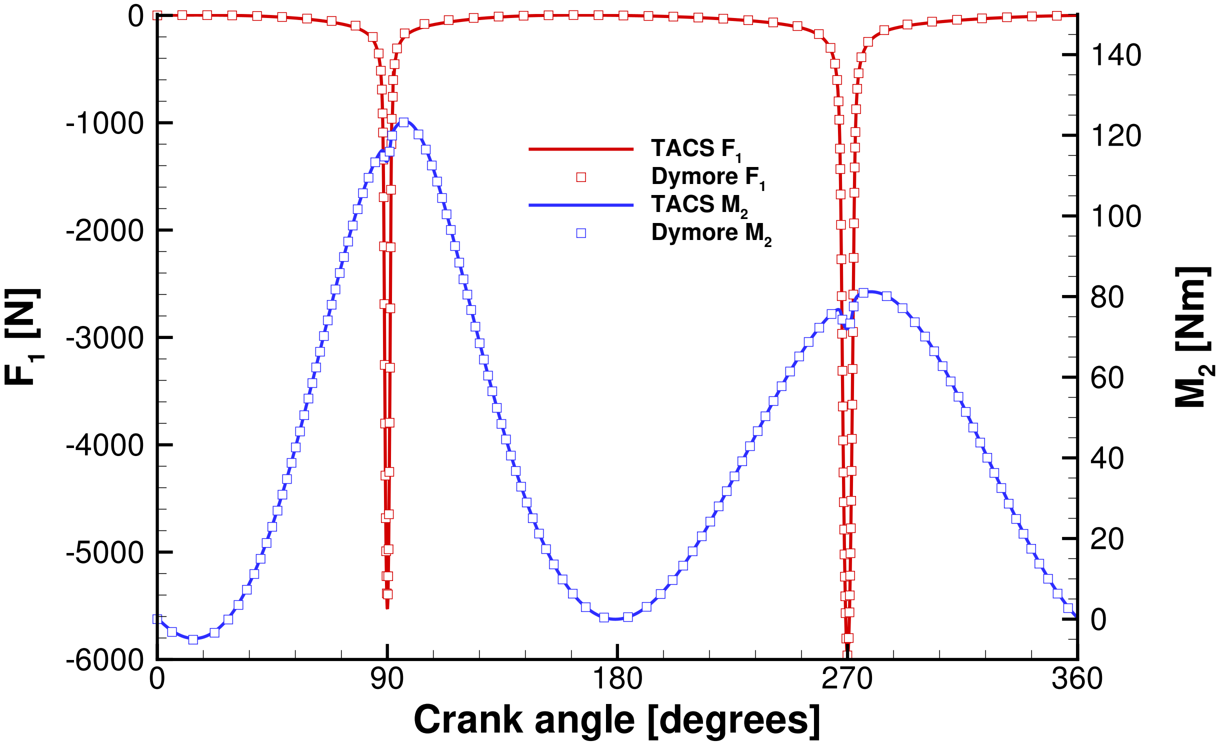


*Figure 5.11: Comparison of TACS and Dymore [1] predictions of force and bending moment in bar AB at mid-span.*

## 5.4 Rotorcraft Hub Dynamics

Typical rotorcraft hub assemblies fall under teetering, fully-articulated, hingeless and bearingless categories. These types differ in the mechanism used to achieve desired flight modes, such as hover or forward flight, and maneuvers, such as roll, pitch and yaw. To achieve these desired flight modes, the control mechanism must impart collective and cyclic control inputs to the blades through the swashplate driven by the push rods. The hub and control chain dynamics are a central part of the rotorcraft flight control system and must be accurately modeled to achieve good performance prediction.

### 5.4.1 Model Description

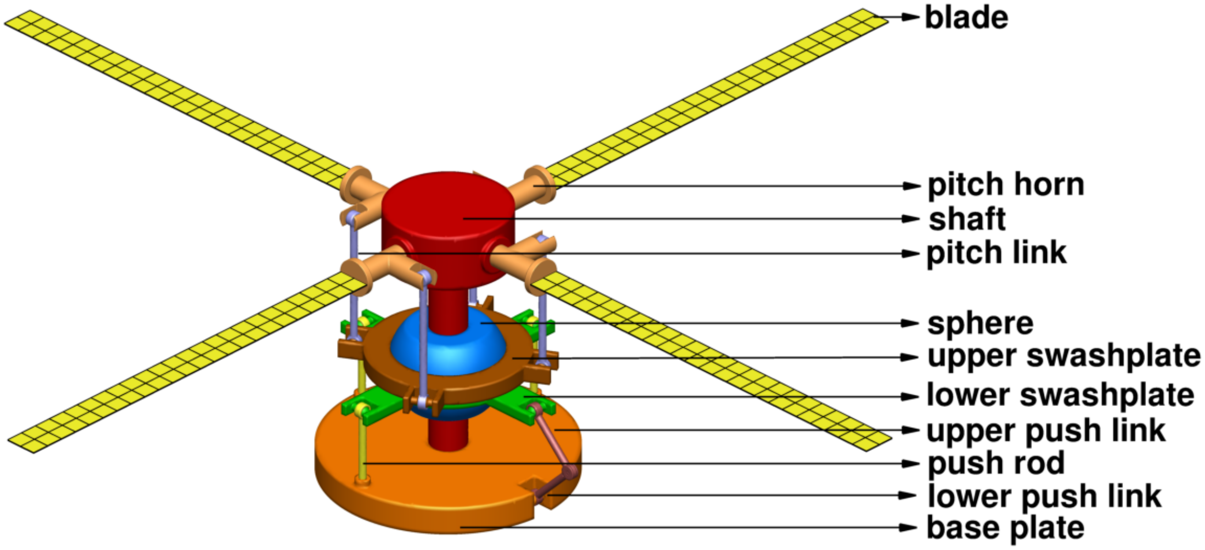


*Figure 5.12: The baseline structural model of rotorcraft hub assembly with its parts labeled.*

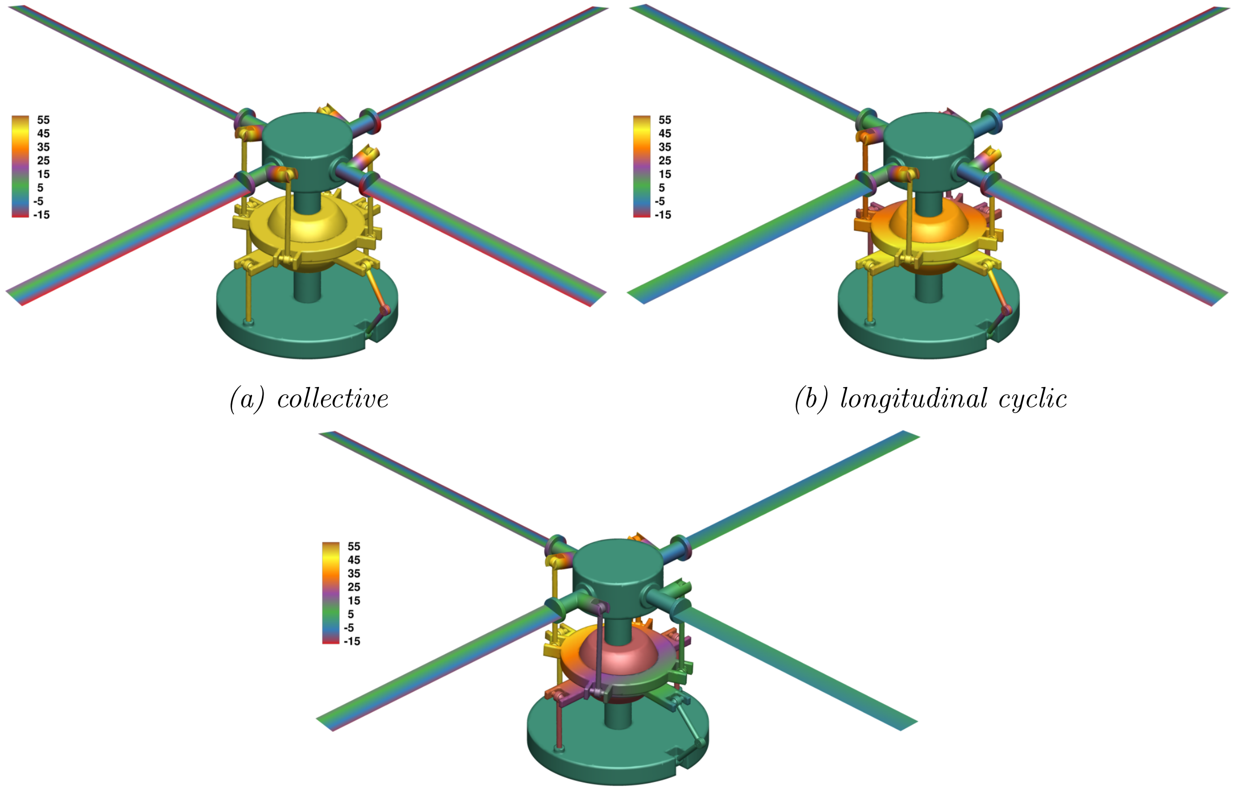
The control chain used for changing the pitch of rotor blades via appropriate inputs to the swashplate is investigated as the motion of interest. The representative four-bladed rotor hub assembly model used for this application is shown in Figure 5.12. The model consists of rigid bodies, kinematic constraints, flexible bodies and actuators. The four rotor blades are modeled as flexible using shell elements whereas the remaining parts are modeled as rigid bodies. The kinematic constraints and actuators used in the rotor assembly are listed in Table 5.4. The push rods are connected to translational actuators that feed time dependent periodic motions, given by $u(t) = u_0 \sin(\Omega_t t)$, where $\Omega_t$ is the assumed translational control signal frequency. This driven motion will be used as the basis for the study of different rotorcraft simulation scenarios in the examination of the rotor hub dynamics. The central shaft is connected to a rotational driver with a angular rate of $\Omega_r = 109.12\,\text{rad/s}$. This structural model contains a total of $28,640$ degrees of freedom. The geometric modeling and meshing parametrization of rotor hub parts is performed using the open-source program GMSH [117]. The inertial properties are obtained directly from the geometry of each part.

*Table 5.4: List of constraint types and motion actuators associated with different bodies in hub assembly model.*

| **Constraint/Actuator** | **Part 1** | **Part 2** |
|---|---|---|
| Rotational actuator | shaft | – |
| Translational actuator | push rod | – |
| Spherical constraint | lower swashplate | sphere |
| Spherical constraint | upper swashplate | pitch link |
| Spherical constraint | pitch link | pitch horn |
| Spherical constraint | lower swashplate | pitch rod |
| Spherical constraint | lower swashplate | upper push link |
| Revolute constraint | lower swashplate | upper swashplate |
| Revolute constraint | shaft | pitch horn |
| Revolute constraint | baseplate | lower push link |
| Revolute constraint | lower push link | upper push link |
| Cylindrical constraint | sphere | shaft |
| Fixed constraint | baseplate | – |

#### *5.4.1.1 Dynamics*

The rotor hub apparatus is studied for $(a)$ collective $(b)$ longitudinal cyclic and $(c)$ lateral cyclic pitch control imparted through the three push rods at $90°, 180°$ and $270°$ from a horizontal reference axis. These conditions are summarized in Table 5.5. The corresponding



*(a) collective*

*(b) longitudinal cyclic*

*(c) lateral cyclic*

*Figure 5.13: Contour plots showing the vertical displacement of bodies during the motion in millimeters.*

time evolution of the configuration of the rotor assembly is simulated using two-stage, third-order diagonally implicit Runge–Kutta method employing a time step size corresponding to 1° per step at the angular rate 109.12 rad/s. Figure 5.13 presents the hub assembly at different time instances for each flight scenario listed in Table 5.5. The contours illustrate the tilting of the swashplate mechanism that produces a pitching motion for each of the blades. In the collective case, the blades attain equal blade pitch, which would produce a net upward aerodynamic force during flight. In the longitudinal and lateral cyclic cases, the pitch of the blades varies as a function of the azimuthal angle and would produce longitudinal and lateral aerodynamic moments during flight. Therefore, this model combined with the control actuation inputs, can represent different flight scenarios and readily lends itself to a multiscenario optimization case which will be demonstrated later in this section.

*Table 5.5: Sinusoidally modulated control amplitudes supplied to the push rods to produce different flight scenarios.*

| **Control** | **Motion** | **Push rod 1** | **Push rod 2** | **Push rod 3** |
|---|---|---|---|---|
| Collective blade pitch control | vertical | $0.050\sin(\Omega_t t)$ | $0.050\sin(\Omega_t t)$ | $0.050\sin(\Omega_t t)$ |
| Longitudinal cyclic blade pitch control | forward/pitch | $0.025\sin(\Omega_t t)$ | $0.025\sin(\Omega_t t)$ | $0.050\sin(\Omega_t t)$ |
| Lateral cyclic blade pitch control | sideways/roll | $0.025\sin(\Omega_t t)$ | $0.050\sin(\Omega_t t)$ | $0.025\sin(\Omega_t t)$ |

### 5.4.2 Adjoint Gradient Verification

Figure 5.14 shows the absolute difference between the adjoint derivatives and the complex-step derivatives on the vertical axis for twelve test functionals indexed on the horizontal axis, for different orders of the DIRK time integration method. The functionals used for this verification are the structural mass (index 1), the average structural compliance (index 2), the KS aggregates of the von Mises failure criterion (indices 3 and 4), and the induced exponential aggregates [46] of the von Mises failure criterion (indices 5 to 12). Table 5.6 shows the magnitude of discrete adjoint sensitivities along with corresponding complex-step sensi-

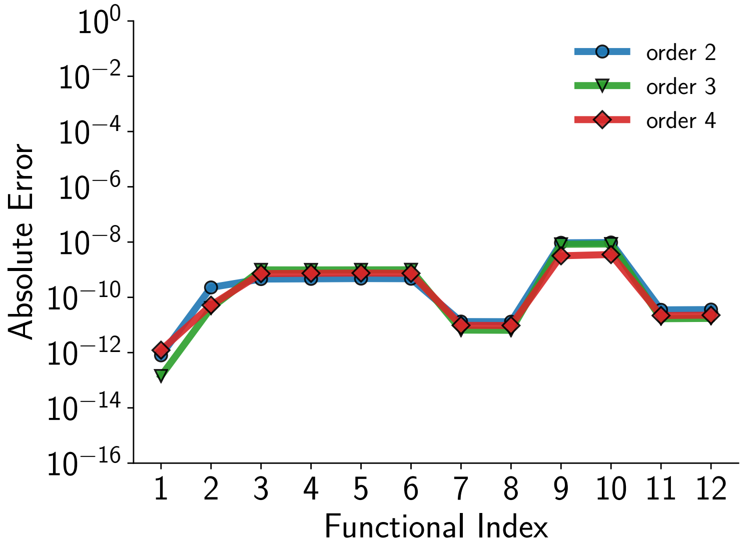


*Figure 5.14: Complex-step verification of the DIRK adjoint formulation of different orders of accuracy with 12 functionals with a perturbation size $\delta = 10^{-16}$.*

tivities. The digits in boldface represent the entries differing from the complex-step method. From Table 5.6 and Figure 5.14 it can be seen that the adjoint-based derivative exhibits an accuracy of 8 to 14 significant digits for different functionals. Thus, the adjoint-based gradient evaluation capability achieves sufficient accuracy for gradient-based optimization.

*Table 5.6: Comparison of complex-step and discrete adjoint derivatives for fourth-order DIRK with a perturbation size $\delta = 10^{-16}$.*

| **Functional** | **Complex-step** | **Adjoint** |
|---|---|---|
| Structural Mass | 250.0000000000000 | 249.9999999999999 |
| Compliance | -0.008903780405108 | -0.0089037804**57068** |
| KS (discrete) | -2.510549172940552 | -2.51054917**3663148** |
| KS (continuous) | -2.505178929745161 | -2.5051789**30486006** |
| Induced (exponential) | -2.511154336312865 | -2.51115433**7069819** |
| Induced (discrete exponential) | -2.516692488940226 | -2.51669248**9679126** |
| Induced (discrete exponential squared) | -0.002788026920568 | -0.0027880269**30265** |
| Induced (exponential squared) | -0.002762476355788 | -0.0027624763**65353** |
| Induced (power) | -4.676554025570486 | -4.67655402**8759453** |
| Induced (discrete power) | -4.715417147892726 | -4.7154171**51435161** |
| Induced (power squared) | -0.006574319319522 | -0.0065743193**41451** |
| Induced (discrete power squared) | -0.006679489586803 | -0.006679489**609466** |

### 5.4.3 Rotorcraft Optimization Demonstration

The rotorcraft hub is now subject to gradient-based design optimization including all three analysis scenarios (flight-modes) described above.

#### *5.4.3.1 Optimization Problem*

The objective of this optimization case is to minimize the mass of the of the rotor blades subject to stress constraints such that the von Mises failure ratio aggregated in space and time domains does not exceed 25% of maximum allowable value. A mass objective is chosen since more realistic rotor design objectives require multidisciplinary design criteria that cannot be evaluated without a coupled aeroelastic simulation. The design variables consist of the thicknesses of 48 spanwise panels modeling the rotor blades. Note that the thickness is constant in the chordwise direction. The cross-sectional geometry is held constant throughout the span for this demonstration case. Smoothness requirements are imposed such that thicknesses of successive spanwise panels do not change more than 1 mm. The optimization problem is stated mathematically as follows:

$$\begin{aligned}
&\underset{\xi}{\text{minimize}} && \text{mass} = m(\xi), && \\
&\text{subject to} && \bar{g}^k(\xi) = \xi_k - \xi_{k+1} \leq 1\,\text{mm}, \quad \forall k = 1, \dots 47, && \text{(smoothness requirement)} \\
& && \bar{g}^k(\xi) = \xi_{k+1} - \xi_k \leq 1\,\text{mm}, \quad \forall k = 1, \dots 47, && \text{(smoothness requirement)} \\
& && \bar{g}^k(\xi) = 1 - 4.0\frac{\sigma_{ks}^k}{\sigma_{max}^k} \geq 0, \quad \forall k = 1, \dots 3, && \text{(stress constraint)} \\
&\text{bounds} && 10\,\text{mm} \leq \xi \leq 20\,\text{mm}. &&
\end{aligned} \tag{5.1}$$

Each blade is assigned the same set of design variables so that all blades are identical during design. The time dependent analysis and gradient evaluation for each of the three cases are performed in parallel using five processors. Both the mass objective and the smoothness constraints are independent of the structural state variables and their gradients are obtained analytically using straightforward differentiation. The time dependent adjoint formulation developed in this work is used to evaluate the three stress constraint gradients. The optimization problem with 48 design variables, 3 stress constraints from each flight scenario, and 94 smoothness constraints, is solved using the SLSQP optimizer within the python package pyOpt [118].

#### *5.4.3.2 Optimization Results*

**Optimization History:** Figure 5.15 shows the optimization convergence history. The optimization took 73 iterations starting from an initial design of 2 cm thickness throughout, to converge to infeasibility and optimality tolerance of $10^{-4}$. Note that the mass and stress constraint infeasibilities are normalized with respect to their values at the initial design. The mass of the blades (shown in red) decreases immediately from the starting point and stays nearly constant throughout the rest of optimization. The stress constraint imposed on the lateral blade pitch flight scenario (shown in green) becomes feasible after about 50 iterations. Finally the stress constraints on longitudinal cyclic (shown in orange) and collective pitch (shown in blue) are satisfied near the termination of optimization algorithm. The optimization produced a design that has all three stress constraints active. The optimizer required 222 function evaluations and 88 gradient evaluations. On average, the forward analysis and gradient evaluation took 33 and 8 minutes, respectively, on five processors.

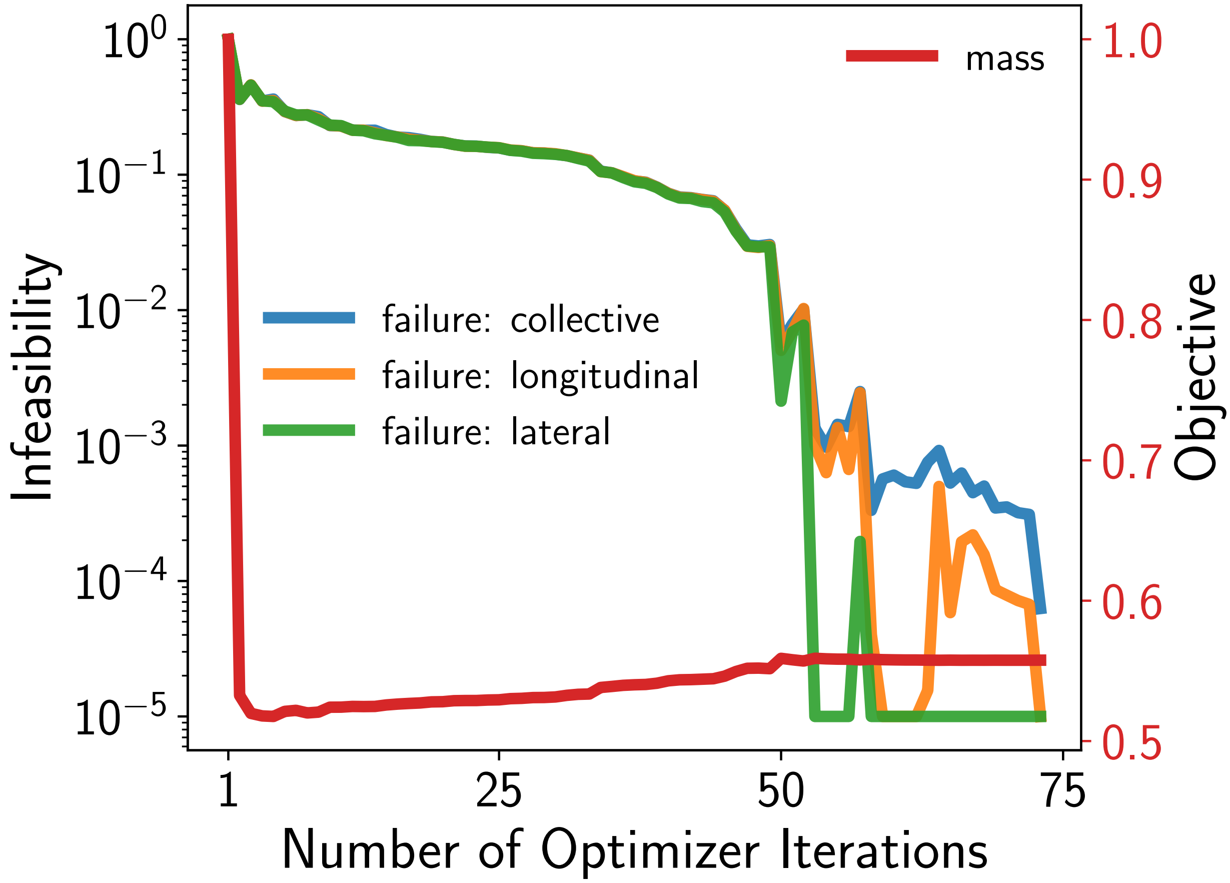


*Figure 5.15: History of optimization showing the changes in normalized constraint infeasibility and objective values.*

**Optimized Design:** The final blade design thickness distribution produced by the optimizer is shown in Figure 5.16. Since the blades have identical design variables, only the optimized thickness from one blade is shown. The optimizer reduced the mass by decreasing

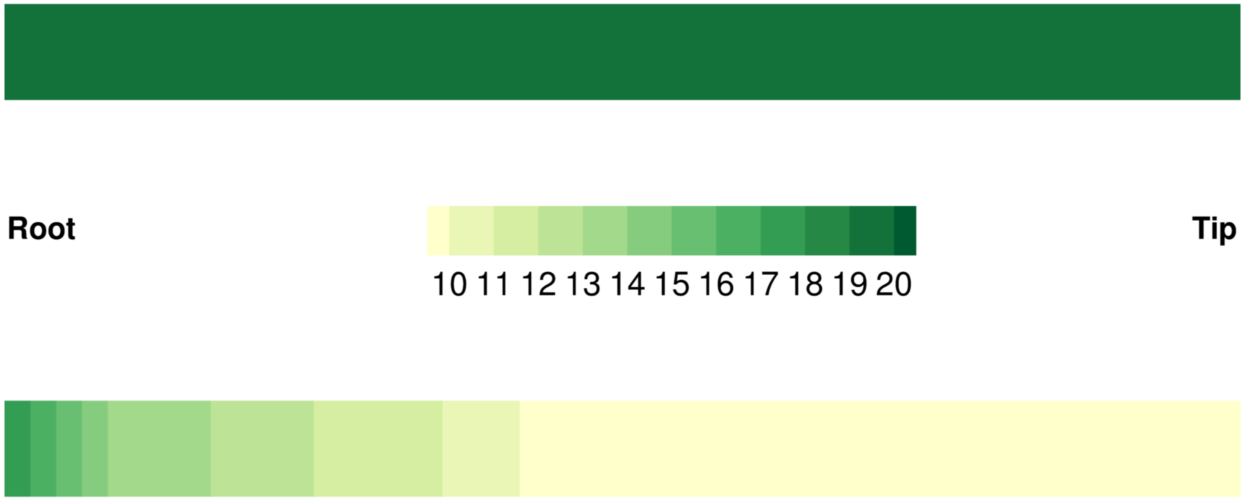
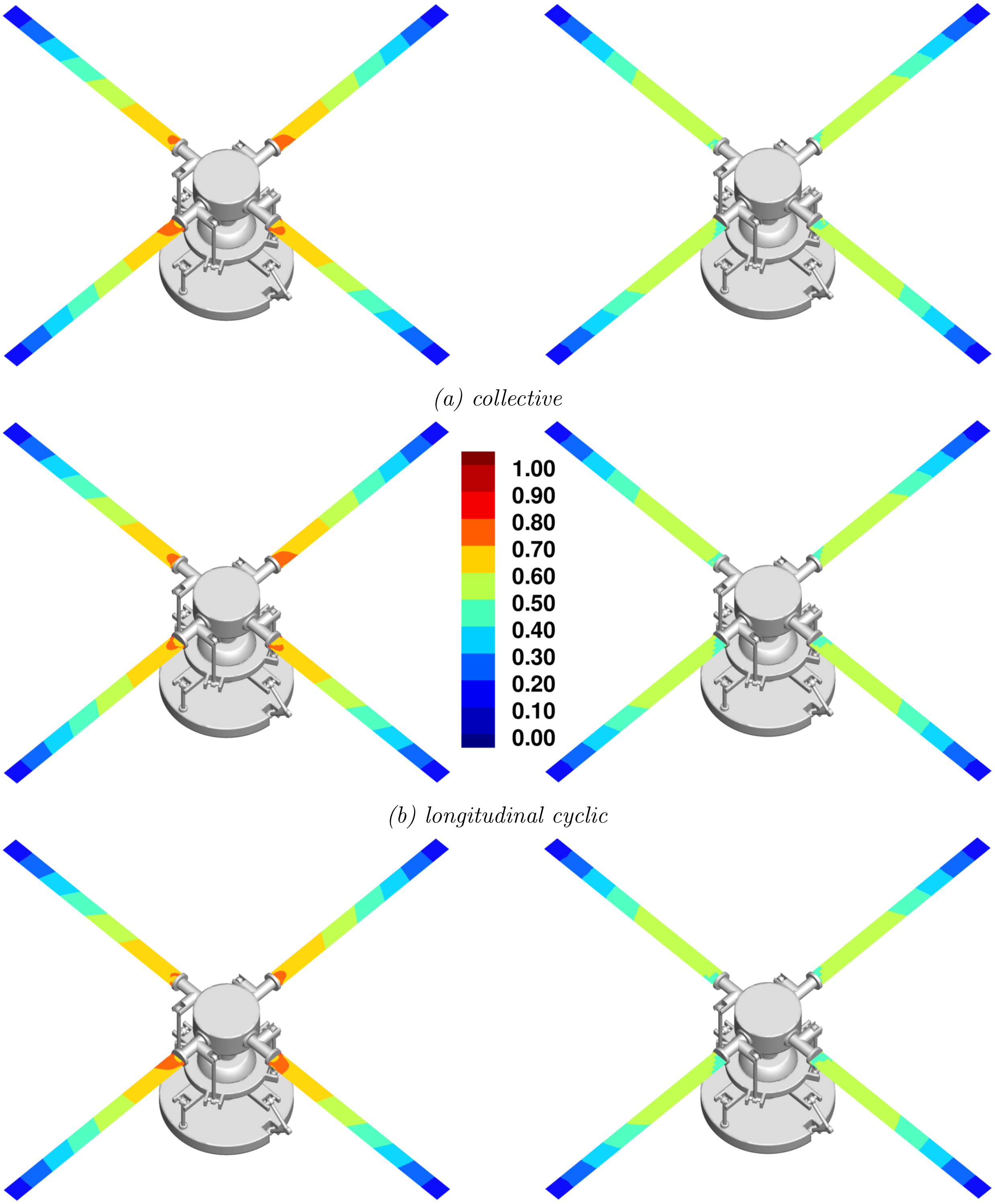


*Figure 5.16: Comparison of thickness of initial (top) and optimized (bottom) designs in millimeters.*

the blade thicknesses towards the tip. The panel thickness gradually decreases along the span until it reaches the lower bound at the tip, thereby reducing the stress at the root. The gradual variation of panel thickness is a result of enforcing the smoothness constraints in the problem formulation. Figure 5.17 compares the instantaneous stress normalized by design stress in the blades after one full rotation. The optimized stresses are lower for all three flight scenarios. The optimizer thickens the shell elements near the root of the blade that experience higher stress to comply with the stress requirements. In addition, there are notable differences in stress distribution patterns between each flight scenario, which is anticipated from the differences in dynamics.

**Summary.** In this chapter, we presented applications of the implicit solution methods and adjoint framework on problems from flexible multibody dynamics. For the rotorcraft optimization application, high-fidelity structural dynamics, efficient sensitivity analysis using the adjoint method, along with the parallel processing capabilities have effectively reduced the optimization time.



*(a) collective*

*(b) longitudinal cyclic*

*(c) lateral cyclic*

*Figure 5.17: Comparison of normalized stress failure ratios of initial (left) and optimized (right) blades for different flight scenarios at 360° azimuth.*

# Part III

# Treatment of Probabilistic Domain in Time Dependent Physical Analysis and Sensitivity Analysis Problems

# CHAPTER 6
# MATHEMATICAL PRELIMINARIES OF UNCERTAINTY ANALYSIS – A LINEAR ALGEBRA APPROACH

**Introduction.** The goal of this chapter is to provide the mathematical preliminaries underlying uncertainty propagation techniques. The chapter is structured such that for each probability distribution type:

1. the transformation of probability spaces

2. the numerical approximation of integrals through quadrature

3. the construction of orthonormal basis functions

are discussed. A detailed review of probability theory is beyond the scope of this work; however, for the presentation of sampling and projection-based uncertainty propagation, this chapter provides sufficient background. A simpler presentation of UQ techniques discussed in this thesis is attributed to the fact that we perform probabilistic computations from a linear algebra perspective using concepts such as *weighted-inner-products*. We lay out this chapter with an interpretation that the probability density functions are the weighting kernel functions of the definition of inner products.

## 6.1 Probability Distribution Functions in Physical and Standard Spaces

Probability distribution functions such as the probability density function (PDF) and the cumulative distribution function (CDF) characterize the distribution of a random variable (denoted as $y$) in stochastic (probabilistic) domain $\mathcal{Y}$. Let us denote the standard stochastic space as $\mathcal{Z}$ and the corresponding standard random variable be denoted as $z$. There is a convention that the standard form of a probability distribution is one that has *location parameter* *as zero* and *scale parameter as one*. Therefore, by shifting and scaling standard distributions we obtain the corresponding general physically-applicable distribution and viceversa. Let us denote the physical stochastic space as $\mathcal{Y}$ and the corresponding physical random variable be denoted as $y$. Although, we should work with physical probability spaces for physical relevance, the standard probabilistic domains are very useful in efficient numerical computations. For instance, numerical tools such as random number generators (to produce random samples), quadrature nodes and weights (to evaluate integrals), orthogonal polynomials (to construct stochastic vector space) are built based on standard probability density functions defined on $\mathcal{Z}$, and simple transformation of variables and bounds can be applied to make these techniques applicable for the physical stochastic space $\mathcal{Y}$. Often, some numerical packages (e.g., Python based `NumPy`) apply different rules for standardization, and provide nodes and weights to evaluate integrals using quadrature. Therefore, in this work, the probabilistic quadrature space is referred to as $\mathcal{X}$ with $x$ as the corresponding variable. An illustration of relation between different probabilistic spaces is shown in Figure 6.1. Therefore, one must be cognizant of the fact that different standardization are used with the derivation of orthonormal polynomials and quadrature libraries, and appropriate transformations ought to be carried out. If quadrature capabilities are solely developed based on standard distributions, we will be able to omit the nonstandard space and work within standardized and physical probability spaces in computations.

### 6.1.1 Uniform Distribution

The uniform distribution of a random variable $y \in [a, b] \subset \mathbb{R}$ is denoted as $\mathcal{U}(y; a, b)$. Its probability density function is the mapping $\rho_u^y(y) : [a, b] \to [0, 1]$ defined as

$$\rho_u^y(y) = \rho_u^y(y; a, b) := \frac{1}{b-a}. \tag{6.1}$$

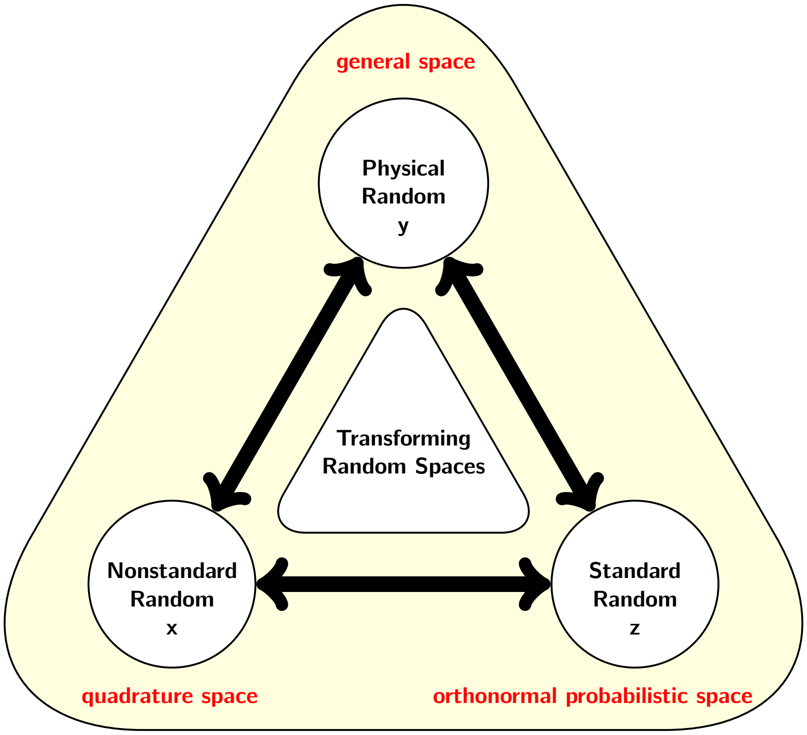


*Figure 6.1: Probabilistic spaces their roles (shown in red) in UQ computations.*

In the case of uniform distribution the transformation between standard uniform $z \in \mathcal{Z}$ and physical uniform random variable $y \in \mathcal{Y}$ is

$$y = a + (b - a)z \tag{6.2}$$

and the corresponding inverse transformation is

$$z = \frac{y - a}{b - a}. \tag{6.3}$$

Note that the standardized stochastic domain is denoted as $\mathcal{Z} = [0, 1]$. The standard uniform probability density function is

$$\rho_u^z(z) = \rho_u^z(z; 0, 1) := 1. \tag{6.4}$$

### 6.1.2 Normal Distribution

Normal or Gaussian distribution of random variable $y \in [-\infty, \infty]$ is denoted as $\mathcal{N}(y; \mu, \sigma^2)$ where $\mu$ and $\sigma^2$ are fixed parameters characterizing the location and stretch of the distribution, and $y$ is the physical random variable. The physical normal probability density function is

$$\rho_n^y(y) = \rho_n^y(y; \mu, \sigma^2) := \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right]. \tag{6.5}$$

The transformation between standard normal $z \in \mathcal{Z}$ and physical normal random variable $y \in \mathcal{Y}$ is

$$y = \mu + \sigma z \tag{6.6}$$

and the corresponding inverse transformation is

$$z = \frac{y-\mu}{\sigma}. \tag{6.7}$$

When the random variable has zero mean and unit variance (a rule for standardization), it results in standard normal PDF

$$\rho_n^z(z) = \rho_n^z(z; 0, 1) := \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}z^2\right). \tag{6.8}$$

Note that $1/\sqrt{2\pi}$ refers to the area under the curve $\exp(-z^2/2)$.

### 6.1.3 Exponential Distribution

We denote an exponential distribution of random variable $y \in [\mu, \infty) \subset \mathbb{R}$ as $\mathcal{E}(y; \mu, \beta)$. Its probability density function is the mapping $\rho_e^y(y) : [\mu, \infty) \to [0, 1]$ defined as

$$\rho_e^y(y) = \rho_e^y(y; \mu, \beta) := \frac{1}{\beta} \exp\left[-\left(\frac{y-\mu}{\beta}\right)\right]. \tag{6.9}$$

The standard exponential distribution pertains to $\mu = 0$ (location) and $\beta = 1$ (scale). The standard exponential probability density function is

$$\rho_e^z(z) = \rho_e^z(z; 0, 1) := \exp(-z). \tag{6.10}$$

The transformation between standard exponential $z \in \mathcal{Z}$ and physical exponential random variable $y \in \mathcal{Y}$ is

$$y = \mu + \beta z \tag{6.11}$$

and the corresponding inverse transformation is

$$z = \frac{y - \mu}{\beta}. \tag{6.12}$$

Note that the standardized stochastic domain is denoted as $\mathcal{Z} = [0, \infty)$.

**Summary.** The PDFs introduced here play the role of weighting kernel functions used in inner products. We recall that inner products are evaluated using integrals, which in turn are numerically approximated using quadrature rules. The domain (independent-axis) of the kernel functions acts as the lower and upper bound on the integrals. Note that the domain (or limits) is not always from $[-\infty, +\infty]$, and depends on the distribution type. Therefore, one must exercise caution in evaluating these integrals, particularly when more random variables are present, with each variable (probabilistic dimension) possibly having different domains.

### 6.2 Statistical Measures as Inner Products

Now, we formally define the probabilistic measures of interest that are used in the formulation of optimization under uncertainty problem (2.17).

### 6.2.1 Probability

The PDFs are useful in evaluating the probability of occurrence of random events. If one considers a random event as the occurrence of random variable within the interval $[a, b]$, then the probability of occurrence of the random event is

$$\mathbb{P}[a \leq y \leq b] = \Big\langle 1 \,\Big|\, 1 \Big\rangle^{\mathcal{Y}}_{\rho^y(y)} = \int_a^b \rho^y(y)\ dy. \tag{6.13}$$

where $\rho^y(y)$ is the probability density function of the random variable $y \in \mathcal{Y}$. This probability is the area under the PDF within the lower and upper limits of integration.

### 6.2.2 First Moment: Mean/Expectation

If the distribution of random variable $y \in \mathcal{Y}$ admits a probability density function $\rho^y(y)$, then the expectation of $y$ is defined as

$$\mathbb{E}[y] = \Big\langle 1 \,\Big|\, y \Big\rangle^{\mathcal{Y}}_{\rho^y(y)} = \int_{\mathcal{Y}} y\ \rho^y(y)\, dy. \tag{6.14}$$

Similarly, we can define the expectation (first moment) of a function of random variable $f(y, \cdot)$ (referred to as random function) as

$$\mathbb{E}[f(y, \cdot)] = \Big\langle 1 \,\Big|\, f(y, \cdot) \Big\rangle^{\mathcal{Y}}_{\rho^y(y)} = \int_{\mathcal{Y}} f(y, \cdot)\ \rho^y(y)\, dy. \tag{6.15}$$

**Remark on the role of random variables and functions in differential equations:** Differential equations often contain unknown parameters that are specified as inputs to run the numerical solution method to solve for the state variables. To accurately represent certain physical scenarios, we model the unknown parameters to be a function of probabilistic random variables. As a consequence, the state functions acquire the dependence of random variables and become probabilistic random functions. It is of use to refer to Figure 2.12 outlining the solution process in abstract form, where $\xi$ refers to the input parameters that emerge from the probabilistic domain under the premise that they are uncertain and dependent on random variable $y$ with some known probability distribution. As a result we obtain the random state functions such as $u(y(\xi), \cdot)$, where $\cdot$ signifies the possible presence/absence of other domains (e.g., temporal, spatial).

### 6.2.3 Second Moment: Variance

If the distribution of random variable $y \in \mathcal{Y}$ admits a probability density function $\rho^y(y)$, then the variance (second moment) of $y$ is defined as

$$\begin{aligned}\mathbb{V}[y] &= \left\langle (\mathbb{E}[y] - y) \;\middle|\; (\mathbb{E}[y] - y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ &= \int_{\mathcal{Y}} (\mathbb{E}[y] - y)^2 \, \rho^y(y) \, dy.\end{aligned} \tag{6.16}$$

With algebraic simplifications, the variance of $y$ can be written equivalently as

$$\mathbb{V}[y] = \mathbb{E}[y^2] - \mathbb{E}[y]^2. \tag{6.17}$$

Similarly, variance of a <u>random function</u> $f(y, \cdot)$ is defined as

$$\begin{aligned}\mathbb{V}[f(y,\cdot)] &= \left\langle \mathbb{E}[f(y,\cdot)] - f(y,\cdot) \;\middle|\; \mathbb{E}[f(y,\cdot)] - f(y,\cdot) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ &= \int_{\mathcal{Y}} (\mathbb{E}[f(y,\cdot)] - f(y,\cdot))^2 \rho^y(y) \, dy.\end{aligned} \tag{6.18}$$

With algebraic simplifications, the variance of $f(y, \cdot)$ can be written equivalently as

$$\mathbb{V}[f(y,\cdot)] = \mathbb{E}[f(y,\cdot)^2] - \mathbb{E}[f(y,\cdot)]^2. \tag{6.19}$$

If the above integrals/inner products could not be evaluated by analytical means, quadrature rules discussed in Section 6.4 are useful in obtaining numerical estimates.

## 6.3 Types of Random Variables

### 6.3.1 Independent Random Variables

Let $y_1$ and $y_2$ be two random variables. They are independent of each other if

$$\mathbb{P}[y_1, y_2] = \mathbb{P}[y_1]\mathbb{P}[y_2]. \tag{6.20}$$

This is a consequence of the independence of probability density functions on each other leading to the construction of the joint PDF as product of individuals PDFs

$$\rho^y(y_1, y_2) = \rho^y_1(y_1) \times \rho^y_2(y_2). \tag{6.21}$$

### 6.3.2 Uncorrelated Random Variables

Let $y_1$ and $y_2$ be two random variables. They are uncorrelated of each other if

$$\mathbb{E}[y_1, y_2] = \mathbb{E}[y_1]\mathbb{E}[y_2] \tag{6.22}$$

Independent random variables are uncorrelated, but uncorrelated random variables are not always independent. In this work, the random variables are assumed to be independent as a simplifying assumption. In the general case, we ought to work with joint probability distributions of the form $\rho^y(y_1, y_2)$ and follow the mathematical steps.

## 6.4 Quadrature Rules for Inner Products in Probabilistic Spaces

For random variable $y$ from physical probabilistic domain $\mathcal{Y}$ with probability density function $\rho^y(y)$, we define inner product as follows

$$\Big\langle g_1(y) \ \Big| \ g_2(y) \Big\rangle^{\mathcal{Y}}_{\rho^y(y)} = \int_{\mathcal{Y}} g_1(y)\rho^y(y)g_2(y)\, dy. \tag{6.23}$$

The probabilistic measures in Section 6.2 are defined using inner products of the form (6.23). Thus the evaluation of inner products of the form (6.23) is of natural interest in stochastic computations.

### 6.4.1 Quadrature Approximation of Inner Products

Quadrature rules are used for numerical approximation of integrals arising from inner products using a finite number of evaluations of the integrand. In stochastic computations, quadrature rules are useful in many scenarios such as:

- the evaluation of probabilities as

$$\left\langle 1(y) \;\middle|\; 1(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} = \int_{\mathcal{Y}} 1(y)\rho^y(y)1(y)\, dy \approx \sum_{i=1}^{M} \alpha_i^y 1(y_i)1(y_i) \tag{6.24}$$

- the evaluation of statistical moments of $f(y)$; for example, the mean is

$$\left\langle 1 \;\middle|\; f(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} = \int_{\mathcal{Y}} 1(y)\rho^y(y)f(y)\, dy \approx \sum_{i=1}^{M} \alpha_i^y 1(y_i)f(y_i). \tag{6.25}$$

- the formation of $(i, j)^{th}$ entry in Jacobian matrices with two corresponding basis elements $\psi_i(y)$ and $\psi_j(y)$ as

$$\left\langle \psi_i(y) \;\middle|\; \psi_j(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} = \int_{\mathcal{Y}} \psi_i(y)\rho^y(y)\psi_j(y)\, dy \approx \sum_{i=1}^{M} \alpha_i^y \psi_i(y_i)\psi_j(y_i). \tag{6.26}$$

The effect of $\rho^y(y)$ is accounted within the weights $\alpha^y$, and by our construction the sum of weights is *unity*: $\sum_{i=1}^{M} \alpha_i^y = 1$. The number of quadrature points $M$ can be determined based on the polynomial degree, $d$, of the full integrand. We may use the fact that quadrature rules with $M$ nodes exactly evaluates polynomial integrands of degree upto $2d - 1$. On the other hand, when the integrand $f(y)$ does not admit a known functional form, the number of evaluations $M$ is usually chosen based on computational budget considerations or iterative convergence criteria. The weight $\alpha_i^y$ and nodes $y_i$ are chosen based on probability distribution of random variable $y \in \mathcal{Y}$.

### 6.4.2 Normal Distribution : Gauss–Hermite Quadrature

Consider two random functions $f_1(y)$ and $f_2(y)$ where $y \sim \mathcal{N}(\mu, \sigma^2) \in \mathcal{Y}$. Using the definition of inner product and transformation of variables

$$\begin{aligned} \left\langle f_1(y) \;\middle|\; f_2(y) \right\rangle_{\rho_n^y(y)}^{\mathcal{Y}} &= \int_{-\infty}^{+\infty} f_1(y) \frac{\exp\left(-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right)}{\sigma\sqrt{2\pi}} f_2(y)\, dy \\ &= \int_{-\infty}^{+\infty} f_1\left(\mu + \sigma z\right) \frac{\exp\left(-\frac{1}{2}\left(z\right)^2\right)}{\sqrt{2\pi}} f_2\left(\mu + \sigma z\right)\, dz \\ &= \left\langle f_1\left(\mu + \sigma z\right) \;\middle|\; f_2\left(\mu + \sigma z\right) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} \end{aligned} \tag{6.27}$$

Let $y$ be a normally distributed random variable in domain $\mathcal{Y}$ with mean $\mu$ and standard deviation $\sigma$. Gauss-Hermite quadrature rule provides nodes $y_i$ and weights $\alpha_i^y$ for optimal numerical approximation of integrals such as

$$\int_{-\infty}^{\infty} f(y) \underbrace{\frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}}_{\rho_n^y(y)}\, dy \approx \sum_{i=1}^{M} \alpha_i^y f(y_i). \tag{6.28}$$

where $\rho_n^y(y)$ is the Gaussian probability density function.

#### *6.4.2.1 Transforming Nonstandard to Physical Quadrature*

Gauss–Hermite quadrature approximates inner products/integrals such as

$$\left\langle 1(\xi) \;\middle|\; f(\xi) \right\rangle_{\rho_n^\xi(\xi)}^{\mathcal{X}} = \int_{-\infty}^{\infty} \underbrace{\mathrm{e}^{-\xi^2}}_{\rho_n^\xi(\xi)} f(\xi)\, d\xi \approx \sum_{i=1}^{M} \alpha_i^\xi f(\xi_i) \tag{6.29}$$

and provide corresponding nodes $\xi_i$ and weights $\alpha_i^\xi$ – which can not be directly applied to inner products/integrals arising in physical probabilistic domain $\mathcal{Y}$. In this case, notice that

$\xi \in \mathcal{X} = [-\infty, \infty]$ is the same domain of integration in (6.28), but a nonstandard density function of $\exp(-\xi^2)$ used to derive the nodes $\xi_i$ and weights $\alpha_i^\xi$. Although we have obtained the identities relating $\mathcal{Y}$ and $\mathcal{Z}$, we are left with the task of identifying transformation from nonstandard space $\mathcal{X}$ to standard space $\mathcal{Y}$. For this, let us perform transformation of variables by defining

$$\xi = \frac{y-\mu}{\sigma\sqrt{2}} \quad \text{and} \quad d\xi = \frac{dy}{\sigma\sqrt{2}}. \tag{6.30}$$

Rearranging this, we get

$$y = \mu + \sigma\sqrt{2}\xi \quad \text{and} \quad dy = \sigma\sqrt{2}d\xi. \tag{6.31}$$

With this transformation (6.28) becomes

$$\begin{aligned} \int_{-\infty}^{\infty} f(y) \frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}\, dy &= \int_{-\infty}^{\infty} f(\mu + \xi\sigma\sqrt{2}) \frac{\mathrm{e}^{-\xi^2}}{\sigma\sqrt{2\pi}}\, (\sigma\sqrt{2})\, d\xi \\ &\approx \frac{1}{\sqrt{\pi}} \sum_{i=1}^{M} \alpha_i^\xi f(\mu + \sigma\sqrt{2}\xi_i). \end{aligned} \tag{6.32}$$

Recognizing the new weights as

$$\alpha_i^y = \alpha_i^\xi / \sqrt{\pi} \tag{6.33}$$

results in the following quadrature form

$$\int_{-\infty}^{\infty} f(y) \underbrace{\frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}}_{\rho_n^y(y)} dy \approx \sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \frac{\alpha_i^\xi}{\sqrt{\pi}} f(\mu + \sigma\sqrt{2}\xi_i). \tag{6.34}$$

Note that the weight $\alpha_i^y$ add up to unity : $\sum_i \alpha_i^y = 1$.

#### *6.4.2.2 Transforming Physical Quadrature to Standard Quadrature*

Let us now determine the weights $\alpha_i^z$ and nodes $z_i$ for performing numerical integration in standard stochastic domain $\mathcal{Z}$. Using the transformation $z = \frac{y-\mu}{\sigma}$ we obtain

$$\int_{-\infty}^{\infty} f(y) \frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}\, dy = \int_{-\infty}^{\infty} f(\mu + \sigma z) \frac{\mathrm{e}^{-\frac{1}{2}(z)^2}}{\sqrt{2\pi}}\, dz. \tag{6.35}$$

Thus the corresponding quadrature approximations follow the relation

$$\sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \alpha_i^z f(\mu + \sigma z). \tag{6.36}$$

We summarize the relations observed from the above development as follows

$$\begin{gathered} \sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \alpha_i^z f(\mu + \sigma z) = \sum_{i=1}^{M} \frac{\alpha_i^\xi}{\sqrt{\pi}} f(\mu + \sigma\sqrt{2}\xi_i) \\ \alpha_i^y = \alpha_i^z = \alpha_i^\xi/\sqrt{\pi} \\ y_i = \mu + \sigma z_i = \mu + \sigma\sqrt{2}\xi_i \end{gathered} \tag{6.37}$$

### 6.4.3 Uniform Distribution : Gauss–Legendre Quadrature

Consider two random functions $f_1(y)$ and $f_2(y)$ where $y \sim \mathcal{U}(a,b) \in \mathcal{Y}$. Using the definition of inner product and transformation of variables

$$\begin{aligned} \Big\langle f_1(y) \;\Big|\; f_2(y) \Big\rangle_{\rho_u^y(y)}^{\mathcal{Y}} &= \int_a^b f_1(y) \left(\frac{1}{b-a}\right) f_2(y)\, dy \\ &= \int_0^1 f_1\left(a + (b-a)z\right)(1) f_2\left(a + (b-a)z\right)\, dz \\ &= \Big\langle f_1\left(a + (b-a)z\right) \;\Big|\; f_2\left(a + (b-a)z\right) \Big\rangle_{\rho_u^z(z)}^{\mathcal{Z}} \end{aligned} \tag{6.38}$$

Let $y$ be a uniformly distributed random variable in domain $\mathcal{Y} = [a,b]$. Gauss-Legendre quadrature rule provides nodes $y_i$ and weights $\alpha_i^y$ for optimal numerical approximation of integrals such as

$$\int_a^b \underbrace{\frac{1}{b-a}}_{\rho_u^y(y)} f(y)\,dy \approx \sum_{i=1}^{M} \alpha_i^y f(y_i). \tag{6.39}$$

#### *6.4.3.1 Transforming Nonstandard to Physical Quadrature*

Numerical implementations of Gauss–Legendre quadrature assume the form

$$\Big\langle 1(\xi) \,\Big|\, f(\xi) \Big\rangle_{\rho_u^\xi(\xi)}^{\mathcal{X}} = \int_{-1}^{1} \underbrace{(1)}_{weight} f(\xi)\,d\xi \approx \sum_{i=1}^{M} \alpha_i^\xi f(\xi_i). \tag{6.40}$$

The domain of integration $\mathcal{X} = [-1, 1]$ is different than $\mathcal{Y} = [a, b]$. Let us scale and shift the nonstandard probabilistic space $\mathcal{X}$ as follows

$$y = \underbrace{\frac{b+a}{2}}_{\text{shift}} + \underbrace{\frac{b-a}{2}}_{\text{scale}}\,\xi \quad \text{and} \quad dy = \frac{b-a}{2}\,d\xi. \tag{6.41}$$

Using the above transformation in (6.39) results in

$$\begin{aligned} \int_a^b \underbrace{\frac{1}{b-a}}_{\rho_u^y(y)} f(y)\,dy &= \int_{-1}^{+1} \frac{1}{b-a} f\left(\frac{b+a}{2} + \frac{b-a}{2}\xi\right) \frac{b-a}{2}\,d\xi \\ &\approx \frac{1}{2}\sum_{i=1}^{M} \alpha_i^\xi f\left(\frac{b+a}{2} + \frac{b-a}{2}\xi_i\right) \end{aligned} \tag{6.42}$$

Comparing with (6.39) we recognize the new weights as

$$\alpha_i^y = \alpha_i^\xi/2 \tag{6.43}$$

which leads to the following discrete approximation for the original integral

$$\int_a^b \underbrace{\frac{1}{b-a}}_{\rho_u^y(y)} f(y)\,dy \approx \sum_{i=1}^{M} \alpha_i^y f(y_i). \tag{6.44}$$

#### *6.4.3.2 Transforming Physical Quadrature to Standard Quadrature*

Let us now determine the weights $\alpha_i^z$ and nodes $z_i$ for performing numerical integration in standard stochastic domain $\mathcal{Z}$ for uniform distribution in $[a, b]$. Using the transformation $z = \frac{y-a}{b-a}$ we obtain

$$\int_a^b f(y)\ \frac{1}{b-a}\,dy = \int_0^1 f(a+(b-a)z)(1)\,dz. \tag{6.45}$$

Thus the corresponding quadrature approximations follow the relation

$$\sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \alpha_i^z f(a+(b-a)z). \tag{6.46}$$

We summarize the relations observed from the above development as follows

$$\begin{gathered}
\sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \alpha_i^z f(a+(b-a)z_i) = \frac{1}{2}\sum_{i=1}^{M} \alpha_i^\xi f\left(\frac{b+a}{2} + \frac{b-a}{2}\xi_i\right) \\
\alpha_i^y = \alpha_i^z = \alpha_i^\xi/2 \\
y_i = a+(b-a)z_i = \frac{b+a}{2} + \frac{b-a}{2}\xi_i
\end{gathered} \tag{6.47}$$

### 6.4.4 Exponential Distribution : Gauss–Laguerre Quadrature

Consider two random functions $f_1(y)$ and $f_2(y)$ where $y \sim \mathcal{E}(\mu, \beta) \in \mathcal{Y}$. Using the definition of inner product and transformation of variables

$$\begin{aligned}
\left\langle f_1(y) \;\middle|\; f_2(y) \right\rangle_{\rho_e^y(y)}^{\mathcal{Y}} &= \int_{\mu}^{+\infty} f_1(y)\frac{1}{\beta}\exp\left(-\frac{y-\mu}{\beta}\right) f_2(y)\,dy \\
&= \int_{-\infty}^{+\infty} f_1\left(\mu+\sigma z\right) \frac{\exp\left(-\frac{1}{2}\left(z\right)^2\right)}{\sqrt{2\pi}} f_2\left(\mu+\sigma z\right)\,dz \\
&= \left\langle f_1\left(\mu+\sigma z\right) \;\middle|\; f_2\left(\mu+\sigma z\right) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}}
\end{aligned} \tag{6.48}$$

#### *6.4.4.1 Transforming Nonstandard to Physical Quadrature*

Gauss–Laguerre quadrature provides approximation for integrals of the form

$$\Big\langle 1(x) \;\Big|\; f(x) \Big\rangle^{\mathcal{X}}_{\rho_e^x(x)} = \int_0^\infty \exp(-x) f(x)\, dx \approx \sum_{i=1}^{M} \alpha_i^x f(x_i). \tag{6.49}$$

The Gauss–Laguerre quadrature provides corresponding nodes $x_i$ and weights $\alpha_i^x$. Let us consider the transformation of the above integral from $\mathcal{X}$ to $\mathcal{Y}$. For this let us perform transformation of variables by defining

$$x = \frac{y-\mu}{\beta} \quad \text{and} \quad dx = \frac{dy}{\beta}. \tag{6.50}$$

Rearranging this, we get

$$y = \mu + \beta x \quad \text{and} \quad dy = \beta dx. \tag{6.51}$$

Using the above transformations, the integral becomes

$$\int_0^\infty \exp(-x) f(x)\, dx = \int_0^\infty \frac{1}{\beta} \exp\left(-\frac{y-\mu}{\beta}\right) f\left(\frac{y-\mu}{\beta}\right) dy \tag{6.52}$$

#### *6.4.4.2 Transforming Physical Quadrature to Standard Quadrature*

For this let us perform transformation of variables by defining

$$z = \frac{y-\mu}{\beta} \quad \text{and} \quad dz = \frac{dy}{\beta}. \tag{6.53}$$

Rearranging this, we get

$$y = \mu + \beta z \quad \text{and} \quad dy = \beta dz. \tag{6.54}$$

Using the above transformations, the integral becomes

$$\int_0^\infty \exp(-z) f(z)\, dz = \int_0^\infty \frac{1}{\beta} \exp\left(-\frac{y-\mu}{\beta}\right) f\left(\frac{y-\mu}{\beta}\right) dy \tag{6.55}$$

We summarize the relations observed from the above development as follows

$$\begin{gathered}\sum_{i=1}^{M} \alpha_i^y f(y_i) = \sum_{i=1}^{M} \alpha_i^z f(\mu + \beta z_i) = \sum_{i=1}^{M} \alpha_i^x f\left(\mu + \beta x_i\right) \\ \alpha_i^y = \alpha_i^z = \alpha_i^x \\ y_i = \mu + \beta z_i = \mu + \beta x_i\end{gathered} \tag{6.56}$$

## 6.5 Orthonormal Polynomials as Basis Functions

Our ultimate goal is to construct the probabilistic space $\mathcal{Y}$ using a set of $N$ basis functions as

$$\mathcal{Y} = \mathrm{span}\{\widehat{\psi}_0^y(y), \widehat{\psi}_1^y(y), \ldots, \widehat{\psi}_N^y(y)\}. \tag{6.57}$$

Here, the notation $\widehat{\cdot}$ represents the orthogonality and normality of polynomials, similar in purpose to Cartesian unit vectors $\widehat{e}_x, \widehat{e}_y$, and $\widehat{e}_z$, and the superscript denotes the variable $y$ that acts as argument. There are numerous possibilities for the choice of basis functions and their corresponding supports (local and global supports). We consider only polynomial functions with global support in this work. Although we can select any set of independent basis functions (for example $\mathcal{Y} = \mathrm{span}\{1, y, y^2, \ldots, y^N\}$), by construction we intend to make the basis functions $\widehat{\psi}^y(y)$ orthonormal to one another, which turns out to be efficient when decomposing (comparing) two stochastic functions, say $f(y)$ and $g(y)$ from the probabilistic domain with PDF as the weight. The decomposition coefficients are directly obtained in an orthonormal basis compared to a non-orthonormal basis where a system of equations needs to be solved to obtain such coefficients. Orthonormality of two basis functions $\widehat{\psi}_i^y(y)$ and

$\widehat{\psi}_j^y(y)$ is mathematically written as

$$\left\langle \widehat{\psi}_i^y(y) \,\middle|\, \widehat{\psi}_j^y(y) \right\rangle_{w^y(y)}^{\mathcal{Y}} = \int_{\mathcal{Y}} \widehat{\psi}_i^y(y) w^y(y) \widehat{\psi}_j^y(y)\ dy = \begin{cases} 1 & \text{if } i = j \\ \\ 0 & \text{if } i \neq j \end{cases}, \tag{6.58}$$

where $w^y(y) \geq 0$ for $y \in \mathcal{Y}$ is a non-zero weighting function in probabilistic domain $\mathcal{Y}$. The inner product (6.58) can be interpreted as a *weighted-inner product* corresponding to a positive definite weight function $w^y(y)$. If $w^y(y) = 1$ we get the standard inner product. The probability density functions (PDF) (denoted as $\rho^y(y)$) are very useful choice for weight functions as they are positive-define. The orthonormal polynomial set is derived for standard distributions and transformation of variables is used to apply them for physical domain. The standardization is usually done with zero location and unit stretch as distribution parameters. The PDFs used in this work are listed in Table 6.1.

*Table 6.1: Physical and standard probability density functions.*

| Distribution | Notation | $\rho^y(y)$ | Standardization | $\rho^z(z)$ |
|---|---|---|---|---|
| Gaussian/Normal | $\mathcal{N}(y; \mu, \sigma)$ | $\frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right]$ | $z = \frac{y-\mu}{\sigma}$ | $\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}z^2\right)$ |
| Uniform | $\mathcal{U}(y; a, b)$ | $\frac{1}{b-a}$ | $z = \frac{y-a}{b-a}$ | $1$ |
| Exponential | $\mathcal{E}(y; \mu, \beta)$ | $\frac{1}{\beta} \exp\left[-\left(\frac{y-\mu}{\beta}\right)\right]$ | $z = \frac{y-\mu}{\beta}$ | $\exp(-z)$ |

### 6.5.1 Orthonormal Hermite Polynomials

Let $z$ be the standardized normal random variable $z = \frac{y-\mu}{\sigma}$. The applicable weight function is the standardized probability density function (6.8). The first five Hermite polynomials are

$$
\begin{aligned}
\overline{H}_0^z(z) &= 1 \\
\overline{H}_1^z(z) &= z \\
\overline{H}_2^z(z) &= z^2 - 1 \\
\overline{H}_3^z(z) &= z^3 - 3z \\
\overline{H}_4^z(z) &= z^4 - 6z^2 + 3
\end{aligned}
\tag{6.59}
$$

These Hermite polynomials follow two term recursive relation

$$
\overline{H}_d^z(z) = z\ \overline{H}_{d-1}^z(z) - (d-1)\overline{H}_{d-2}^z(z) \tag{6.60}
$$

where $d$ is the degree of the polynomial.

**Orthogonality of Hermite polynomials.** It is easy to see that these Hermite polynomials are orthogonal to each other with Gaussian PDF as the weight. For example

$$
\begin{aligned}
\left\langle \overline{H}_0^z(z) \,\middle|\, \overline{H}_1^z(z) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} &= \int_{-\infty}^{\infty} (1) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right)^{z} (z)\ dz = 0 \\
\left\langle \overline{H}_1^z(z) \,\middle|\, \overline{H}_2^z(z) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} &= \int_{-\infty}^{\infty} (z) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (z^2 - 1)\ dz = 0 \\
\left\langle \overline{H}_3^z(z) \,\middle|\, \overline{H}_2^z(z) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} &= \int_{-\infty}^{\infty} (z^3 - 3z) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (z^2 - 1)\ dz = 0
\end{aligned}
\tag{6.61}
$$

This is due to the symmetry of the integrand with respect to the horizontal $z$-axis.

**Normality of Hermite polynomials.** The squared length (norm) of each polynomial can be found as

$$
\begin{aligned}
\left\langle \overline{H}^z_0(z) \,\middle|\, \overline{H}^z_0(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} &= \int_{-\infty}^{\infty} (1) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (1)\; dz = 0 = 0! \\
\left\langle \overline{H}^z_1(z) \,\middle|\, \overline{H}^z_1(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} &= \int_{-\infty}^{\infty} (z) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right)^{z} (z)\; dz = 1 = 1! \\
\left\langle \overline{H}^z_2(z) \,\middle|\, \overline{H}^z_2(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} &= \int_{-\infty}^{\infty} (z^2 - 1) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (z^2 - 1)\; dz = 2 = 2! \\
\left\langle \overline{H}^z_3(z) \,\middle|\, \overline{H}^z_3(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} &= \int_{-\infty}^{\infty} (z^3 - 3z) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (z^3 - 3z)\; dz = 6 = 3! \\
\left\langle \overline{H}^z_4(z) \,\middle|\, \overline{H}^z_4(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} &= \int_{-\infty}^{\infty} (z^4 - 6z^2 + 3) \left( \frac{\exp^{-\frac{z^2}{2}}}{\sqrt{2\pi}} \right) (z^4 - 6z^2 + 3)\; dz = 24 = 4!
\end{aligned}
\tag{6.62}
$$

Thus we have the norm of Hermite polynomial of order $d$ as

$$
\left\langle \overline{H}^z_d(z) \,\middle|\, \overline{H}^z_d(z) \right\rangle^{\mathcal{Z}}_{\rho^z_n(z)} = d! \tag{6.63}
$$

Note that the the basis elements $\overline{H}^z_i(z)$ are not of unit length. Therefore, we divide each element by its length to make it unit length. The resulting orthonormal Hermite polynomials are,

$$
\begin{aligned}
\widehat{H}^z_0(z) &= 1/\sqrt{0!} \\
\widehat{H}^z_1(z) &= z/\sqrt{1!} \\
\widehat{H}^z_2(z) &= (z^2 - 1)/\sqrt{2!} \\
\widehat{H}^z_3(z) &= (z^3 - 3z)/\sqrt{3!} \\
\widehat{H}^z_4(z) &= (z^4 - 6z^2 + 3)/\sqrt{4!}
\end{aligned}
\tag{6.64}
$$

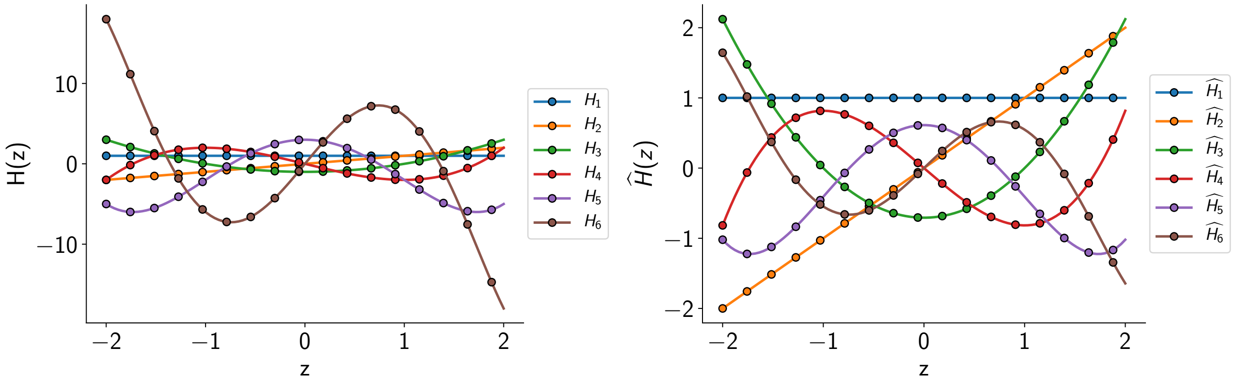
Thus we have the following property about Hermite polynomials depending upon whether they are normalized

$$\left\langle \overline{H}_i^z(z) \,\middle|\, \overline{H}_j^z(z) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} = i! \times \delta_{ij} = \begin{cases} i! & \text{if i = j} \\ 0 & \text{if i} \neq \text{j} \end{cases} \tag{6.65}$$

and

$$\left\langle \widehat{H}_i^z(z) \,\middle|\, \widehat{H}_j^z(z) \right\rangle_{\rho_n^z(z)}^{\mathcal{Z}} = \delta_{ij} = \begin{cases} 1 & \text{if i = j} \\ 0 & \text{if i} \neq \text{j} \end{cases} \tag{6.66}$$

Figure 6.2 shows the first six Hermite polynomials in normalized and unnormalized forms.



*(a) Hermite* *(b) Unit Hermite*

*Figure 6.2: Plot of Hermite and unit Hermite polynomials.*

### 6.5.2 Orthonormal Legendre Polynomials

By using the uniform probability density function $\rho_u^z(z) = 1$ defined in the interval $z \in [0, 1]$ we obtain *Legendre orthogonal polynomials* using Gram-Schmidt process. The affine transformation $z = \frac{y-a}{b-a}$ can be used for transforming physical random variable $y \in [a, b]$ to standard uniform random variable $z \in [0, 1]$. The first five Legendre polynomials are

$$\begin{aligned}
\overline{P}^z_0(z) &= 1 \\
\overline{P}^z_1(z) &= 2z - 1 \\
\overline{P}^z_2(z) &= 6z^2 - 6z + 1 \\
\overline{P}^z_3(z) &= 20z^3 - 30z^2 + 12z - 1 \\
\overline{P}^z_4(z) &= 70z^4 - 140z^3 + 90z^2 - 20z + 1
\end{aligned} \tag{6.67}$$

The Legendre polynomials can be obtained using the relation

$$\overline{P}^z_d(z) = (-1)^d \sum_{k=0}^{d} \begin{pmatrix} d \\ k \end{pmatrix} \begin{pmatrix} d+k \\ k \end{pmatrix} (-z)^k \tag{6.68}$$

where $d$ is the degree of the polynomial.

**Orthogonality of Legendre polynomials.** It is easy to see that these Legendre polynomials are orthogonal to each other when weighed using the standard uniform probability density function $\rho^z_u(z)$. For example

$$\begin{aligned}
\left\langle \overline{P}^z_0(z) \,\middle|\, \overline{P}^z_1(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (1)(1)(2z-1)\ dz = 0 \\
\left\langle \overline{P}^z_1(z) \,\middle|\, \overline{P}^z_2(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (2z-1)(1)(6z^2-6z+1)\ dz = 0 \\
\left\langle \overline{P}^z_3(z) \,\middle|\, \overline{P}^z_2(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (20z^3-30z^2+12z-1)(1)(6z^2-6z+1)\ dz = 0
\end{aligned} \quad . \tag{6.69}$$

This is due to the symmetry of the integrand with respect to the midpoint of the interval $[0, 1]$ which is 0.5.

**Normality of Legendre polynomials.** The squared length (norm) of each Legendre polynomial can be found as

$$\begin{aligned}
\left\langle \overline{P}^z_0(z) \;\middle|\; \overline{P}^z_0(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (1)(1)(1)\ dz = 1 = \frac{1}{2(0)+1} \\
\left\langle \overline{P}^z_1(z) \;\middle|\; \overline{P}^z_1(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (2z-1)(1)(2z-1)\ dz = \frac{1}{3} = \frac{1}{2(1)+1} \\
\left\langle \overline{P}^z_2(z) \;\middle|\; \overline{P}^z_2(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} &= \int_0^1 (6z^2-6z+1)(1)(6z^2-6z+1)\ dz = \frac{1}{5} = \frac{1}{2(2)+1}
\end{aligned} \tag{6.70}$$

Thus we have the norm of Legendre polynomial of order $d$ as

$$\left\langle \overline{P}^z_d(z) \;\middle|\; \overline{P}^z_d(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} = \frac{1}{2(d)+1}. \tag{6.71}$$

Therefore, we can define unit orthogonal Legendre polynomials using normalization

$$\widehat{P}^z_d(z) = \overline{P}^z_d(z)\sqrt{(2(d)+1)}. \tag{6.72}$$

Following the above rule, the first few orthonormal Legendre polynomials are the following

$$\begin{aligned}
\widehat{P^z_0}(z) &= \sqrt{1}(1) \\
\widehat{P^z_1}(z) &= \sqrt{3}(2z-1) \\
\widehat{P^z_2}(z) &= \sqrt{5}(6z^2-6z+1) \\
\widehat{P^z_3}(z) &= \sqrt{7}(20z^3-30z^2+12z-1) \\
\widehat{P^z_4}(z) &= \sqrt{9}(70z^4-140z^3+90z^2-20z+1)
\end{aligned} \tag{6.73}$$

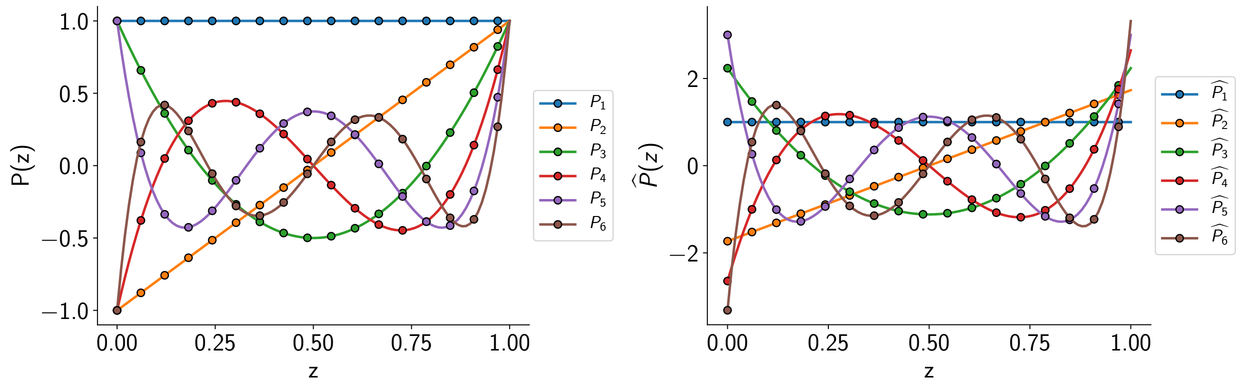
In summary, we have the following property about orthogonal and orthonormal Legendre polynomials

$$\left\langle \overline{P}^z_i(z) \;\middle|\; \overline{P}^z_j(z) \right\rangle^{\mathcal{Z}}_{\rho^z_u(z)} = \frac{1}{2(i)+1} \times \delta_{ij} = \begin{cases} \frac{1}{2(i)+1} & \text{if i = j} \\ 0 \quad \text{if i} \neq \text{j} \end{cases} \tag{6.74}$$

and

$$\left\langle \widehat{P}_i^z(z) \;\middle|\; \widehat{P}_j^z(z) \right\rangle^{\mathcal{Z}}_{\rho_u^z(z)} = \delta_{ij} = \begin{cases} 1 & \text{if i = j} \\ 0 & \text{if i} \neq \text{j} \end{cases} \tag{6.75}$$

Figure 6.3 shows the first six Legendre polynomials in normalized and unnormalized forms.



*(a) Legendre*

*(b) Unit Legendre*

*Figure 6.3: Plot of Legendre and unit Legendre polynomials.*

### 6.5.3 Orthonormal Laguerre Polynomials

The Laguerre polynomials are derived using standard exponential PDF (6.10) as the weight function for orthogonalization. The first five orthogonal Laguerre polynomials are

$$\begin{aligned} \overline{L}_0^z(z) &= +1 \\ \overline{L}_1^z(z) &= -z + 1 \\ \overline{L}_2^z(z) &= +z^2 - 4z + 2 \\ \overline{L}_3^z(z) &= -z^3 + 9z^2 - 18z + 6 \\ \overline{L}_4^z(z) &= +z^4 - 16z^3 + 72z^2 - 96z + 24 \end{aligned} \tag{6.76}$$

**Orthogonality of Laguerre polynomials.** It is easy to see that these Laguerre polynomials are orthogonal to each other. For example

$$\begin{aligned}
\left\langle \overline{L}_0^z(z) \;\middle|\; \overline{L}_1^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (1)\exp(-z)\,(-z+1)\;dz = 0 \\
\left\langle \overline{L}_1^z(z) \;\middle|\; \overline{L}_2^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (-z+1)\exp(-z)\,(+z^2-4z+2)\;dz = 0 \\
\left\langle \overline{L}_3^z(z) \;\middle|\; \overline{L}_2^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (-z^3+9z^2-18z+6)\exp(-z)\,(+z^2-4z+2)\;dz = 0
\end{aligned} \tag{6.77}$$

**Normality of Laguerre polynomials.** The squared length (norm) of each Laguerre polynomial can be found as

$$\begin{aligned}
\left\langle \overline{L}_0^z(z) \;\middle|\; \overline{L}_0^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (1)^2\exp(-z)\;\;dz = 0!^2 \\
\left\langle \overline{L}_1^z(z) \;\middle|\; \overline{L}_1^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (-z+1)^2\exp(-z)\;\;dz = 1!^2 \\
\left\langle \overline{L}_2^z(z) \;\middle|\; \overline{L}_2^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (+z^2-4z+2)^2\exp(-z)\;\;dz = 2!^2 \\
\left\langle \overline{L}_3^z(z) \;\middle|\; \overline{L}_3^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (-z^3+9z^2-18z+6)^2\exp(-z)\;\;dz = 3!^2 \\
\left\langle \overline{L}_4^z(z) \;\middle|\; \overline{L}_4^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} &= \int_0^\infty (+z^4-16z^3+72z^2-96z+24)^2\exp(-z)\;\;dz = 4!^2
\end{aligned} \tag{6.78}$$

Thus we have the norm of Laguerre polynomial of order $d$ as

$$\left\langle \overline{L}_d^z(z) \;\middle|\; \overline{L}_d^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} = d!^2. \tag{6.79}$$

Therefore, we can define unit orthogonal Laguerre polynomials using normalization

$$\widehat{L}_d^z(z) = \frac{\overline{L}_d^z(z)}{d!}. \tag{6.80}$$

Following the above rule, the first few orthonormal Laguerre polynomials are the following

$$\begin{aligned}
\widehat{L}_0^z(z) &= \frac{1}{0!}(1) \\
\widehat{L}_1^z(z) &= \frac{1}{1!}(-z+1) \\
\widehat{L}_2^z(z) &= \frac{1}{2!}(+z^2-4z+2) \\
\widehat{L}_3^z(z) &= \frac{1}{3!}(-z^3+9z^2-18z+6) \\
\widehat{L}_4^z(z) &= \frac{1}{4!}(+z^4-16z^3+72z^2-96z+24)
\end{aligned} \tag{6.81}$$

In summary, we have the following property about orthogonal and orthonormal Laguerre polynomials

$$\left\langle \overline{L}_i^z(z) \,\middle|\, \overline{L}_j^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} = i!^2 \times \delta_{ij} = \begin{cases} i!^2 & \text{if i = j} \\ 0 & \text{if i} \neq \text{j} \end{cases} \tag{6.82}$$

and

$$\left\langle \widehat{L}_i^z(z) \,\middle|\, \widehat{L}_j^z(z) \right\rangle_{\rho_e^z(z)}^{\mathcal{Z}} = \delta_{ij} = \begin{cases} 1 & \text{if i = j} \\ 0 & \text{if i} \neq \text{j} \end{cases} \tag{6.83}$$

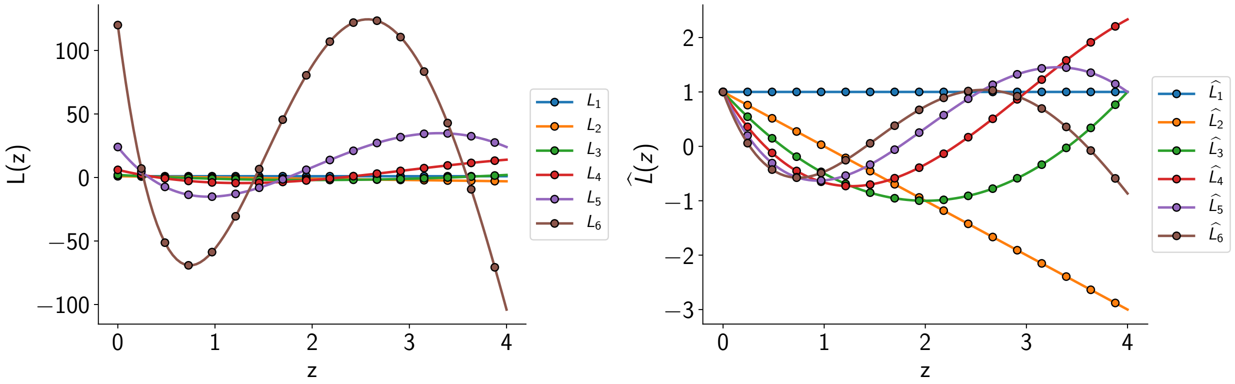
The orthonormal Laguerre polynomials follow two term recursive relation

$$\widehat{L}_d^z(z) = \frac{(2d-1-z)\;\widehat{L}_{d-1}^z(z) - (d-1)\;\widehat{L}_{d-2}^z(z)}{d} \tag{6.84}$$

where $d$ is the degree of the polynomial. If one uses the recursion (6.84), there is no additional normalization necessary. Figure 6.4 shows the first six Laguerre polynomials in normalized and unnormalized forms.

### *6.5.3.1 Orthonormal Basis Set via Gram-Schmidt Process*

Given a set of initial *non-orthonormal* polynomials spanning the standard probabilistic space $\mathcal{Z}$, we construct an *orthogonal* and *orthonormal* set of polynomial basis functions spanning



*(a) Laguerre*

*(b) Unit Laguerre*

*Figure 6.4: Plot of Laguerre and unit Laguerre polynomials.*

the same space $\mathcal{Z}$

$$\mathcal{Z} = \underbrace{\text{span}\{\psi_i(z)\}_{i=1}^{N}}_{initial\ set} = \underbrace{\text{span}\{\overline{\psi}_i(z)\}_{i=1}^{N}}_{orthogonal\ set} = \underbrace{\text{span}\{\widehat{\psi}_i(z)\}_{i=1}^{N}}_{orthonormal\ set} \tag{6.85}$$

This is done using a Gram–Schmidt process. The first step is orthogonalization of initial basis functions which is

$$\overline{\psi}_k^z(z) = \psi_k^z(z) - \sum_{j=1}^{k-1} \left( \frac{\left\langle \psi_k^z(z) \;\middle|\; \overline{\psi}_j^z(z) \right\rangle_{w^z(z)}^{\mathcal{Z}}}{\left\langle \overline{\psi}_j^z(z) \;\middle|\; \overline{\psi}_j^z(z) \right\rangle_{w^z(z)}^{\mathcal{Z}}} \right) \overline{\psi}_j^z(z) \quad \forall k = 1, 2, \ldots, N. \tag{6.86}$$

The resulting orthogonal polynomial functions $\overline{\psi}_k^z(z)$ need not be of unit length. Thus, the second and final step of Gram–Schmidt process is normalization of orthogonal set $\{\overline{\psi}_k^z(z)\}_{k=1}^{N}$ which is

$$\widehat{\psi}_k^z(z) = \overline{\psi}_k^z(z) \Bigg/ \sqrt{\left\langle \overline{\psi}_k^z(z) \;\middle|\; \overline{\psi}_k^z(z) \right\rangle_{w^z(z)}^{\mathcal{Z}}} \quad \forall\ k = 1, 2, \ldots, N. \tag{6.87}$$

Although we can construct an orthonormal set numerically using the above procedure given an initial set of functions and a PDF as weight, it is efficient to derive orthonormal polynomial functions for convenient choices of weighting functions. We summarize the orthonormal polynomials derived using Gram–Schmidt process using probability density functions $\rho^z(z)$ as weighting functions $w^z(z)$ in Table 6.2.

*Table 6.2: Orthonormal polynomials for standard probability distributions.*

| | Hermite | Legendre | Laguerre |
|---|---|---|---|
| notation | $\widehat{H}_d(z)$ | $\widehat{P}_d(z)$ | $\widehat{L}_d(z)$ |
| weight | $\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}z^2\right)$ | $1$ | $\exp(-z)$ |
| distribution | $\mathcal{N}(z;\mu=0,\sigma=1)$ | $\mathcal{U}(z;a=0,b=1)$ | $\mathcal{E}(z;\mu=0,\beta=1)$ |
| orthogonal set | $\overline{H}_d(z)$ | $\overline{P}_d(z)$ | $\overline{L}_d(z)$ |
| 0 | $1$ | $1$ | $1$ |
| 1 | $z$ | $2z-1$ | $-z+1$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $d$ | $z\overline{H}^z_{d-1}(z)-(d-1)\overline{H}^z_{d-2}(z)$ | $(-1)^d\sum_{k=0}^d\binom{d}{k}\binom{d+k}{k}(-z)^k$ | $\frac{(2d-1-z)\ \overline{L}^z_{d-1}(z)}{d}$<br>$-\frac{(d-1)\overline{L}^z_{d-2}(z)}{d}$ |
| normalization | $\widehat{H}^z_d(z)=\overline{H}^z_d(z)/\sqrt{d!}$ | $\widehat{P}^z_d(z)=\overline{P}^z_d(z)\sqrt{2d+1}$ | $\widehat{L}^z_d(z)=\overline{L}^z_d(z)$ |

### 6.5.4 Construction of Multivariate Basis from Univariate Bases

There is no ambiguity in basis construction when there is only one variable : we select polynomials of increasing degree. However, when there are more probabilistic random variables, basis construction can take multiple routes. For this discussion, let us assume that there are two probabilistic random variables. Let $y_1$ be the first probabilistic random variable of assumed degree $d_1$ giving rise to $N_1 = 1 + d_1$ univariate basis entries, $y_2$ be the second probabilistic random variable of assumed degree $d_2$ giving rise to $N_2 = 1 + d_2$.

#### *6.5.4.1 Tensor Product Rule*

If we use the tensor product to construct the bivariate basis $\mathcal{Y} = \mathcal{Y}_1 \otimes \mathcal{Y}_2$ we have $N = N_1 \times N_2 = (1 + d_1) \times (1 + d_2)$ bivariate basis functions of the form $\widehat{\psi}^y(y_1, y_2)$. In general, if there are $M$ probabilistic random variables we will generate

$$N = \prod_{i=1}^{M}(1 + d_i) \tag{6.88}$$

$M-$variate basis functions of the form $\widehat{\psi}^y(y_1, y_2, \ldots, y_M)$. Clearly, the selection of degree for each probabilistic random variables has a strong impact on the number of terms in the entire basis set, and we must be cautious of the associated computational expenses when solving stochastic PDEs.

#### *6.5.4.2 Complete Polynomial Rule*

Let $d_{max} = \max\{d_1, d_2\}$ be the maximum degree among all random variables. We construct the bivariate basis set such that for any basis entry, the sum of degrees does not exceed the maximum degree, that is $d_1 + d_2 \leq d_{max}$. This offers a better construction of basis set compared to tensor product construction.

**Illustrative Example:** Let $y_1 \sim \mathcal{N}(\mu, \sigma)$ with $d_1 = 2$, and $y_2 \sim \mathcal{U}(a, b)$ with $d_2 = 3$. We obtain the following basis set using tensor and complete polynomial based constructions

$$\widehat{\psi}(y_1, y_2) = \left( \begin{array}{ccc} \textbf{basis function} & \textbf{total degree} & \textbf{construction type} \\ \widehat{H}_0^y(y)\widehat{P}_0^y(y) & 0 & \text{tensor, complete} \\ \widehat{H}_0^y(y)\widehat{P}_1^y(y) & 1 & \text{tensor, complete} \\ \widehat{H}_1^y(y)\widehat{P}_0^y(y) & 1 & \text{tensor, complete} \\ \widehat{H}_0^y(y)\widehat{P}_2^y(y) & 2 & \text{tensor, complete} \\ \widehat{H}_1^y(y)\widehat{P}_1^y(y) & 2 & \text{tensor, complete} \\ \widehat{H}_2^y(y)\widehat{P}_0^y(y) & 2 & \text{tensor, complete} \\ \widehat{H}_0^y(y)\widehat{P}_3^y(y) & 3 & \text{tensor, complete} \\ \widehat{H}_1^y(y)\widehat{P}_2^y(y) & 3 & \text{tensor, complete} \\ \widehat{H}_2^y(y)\widehat{P}_1^y(y) & 3 & \text{tensor, complete} \\ \widehat{H}_1^y(y)\widehat{P}_3^y(y) & 4 & \text{tensor} \\ \widehat{H}_2^y(y)\widehat{P}_2^y(y) & 4 & \text{tensor} \\ \widehat{H}_2^y(y)\widehat{P}_3^y(y) & 5 & \text{tensor} \end{array} \right) \tag{6.89}$$

# CHAPTER 7

# ADJOINT BASED OPTIMIZATION UNDER UNCERTAINTY USING PROJECTION AND SAMPLING

> *Because philosophy arises from awe, a philosopher is bound in his way to be a lover of myths and poetic fables. Poets and philosophers are alike in being big with wonder.*
>
> *Thomas Aquinas*

**Introduction.** We are interested in solving OUU problems embedding optimality, robustness and reliability considerations of the form

$$\begin{aligned}
\underset{\xi}{\text{minimize}} \quad & (1-\alpha)\cdot\mathbb{E}[F(\xi,\cdot)] + \alpha\cdot\mathbb{S}[F(\xi,\cdot)] \\
\text{subject to} \quad & \mathbb{E}[G(\xi,\cdot)] + \beta\cdot\mathbb{S}[G(\xi,\cdot)] \leq 0 \\
& \mathbb{E}[H(\xi,\cdot)] = 0 \\
\text{require} \quad & \frac{d\mathbb{E}[F(\xi,\cdot)]}{d\xi}, \frac{d\mathbb{E}[G(\xi,\cdot)]}{d\xi}, \frac{d\mathbb{E}[H(\xi,\cdot)]}{d\xi} \\
& \frac{d\mathbb{S}[F(\xi,\cdot)]}{d\xi}, \frac{d\mathbb{S}[G(\xi,\cdot)]}{d\xi}
\end{aligned} \tag{7.1}$$

Here, the notation $F(\xi,\cdot)$ is the short hand notation for $F(y(\xi), u(y(\xi)), \dot{u}(y(\xi)), \ddot{u}(y(\xi)))$, acknowledging the presence of state variables and their time derivatives. In this chapter, let us explore the computation of the quantities necessary to solve (7.1), that are the probabilistic moments $\mathbb{E}[F]$, $\mathbb{V}[F]$ , $\mathbb{S}[F]$, and design-variable derivatives of probabilistic moments $\dfrac{d\mathbb{E}[F]}{d\xi}$, $\dfrac{d\mathbb{V}[F]}{d\xi}$, $\dfrac{d\mathbb{S}[F]}{d\xi}$. The evaluation of derivatives using the adjoint method has been discussed in Chapters 2, 3, 4, in the context of stationary systems and time dependent systems. In this chapter, the extension of adjoint method for probabilistic systems is presented along the line of sampling and semi-intrusive projection.

## 7.1 Nonintrusive Sampling Method

The principle of nonintrusive methods is to repeatedly evaluate the function of interest at predetermined nodes $y_i$ to compute the probabilistic moments and derivatives. The number of quadrature nodes $M$ is chosen based on the computational budget at hand.

### 7.1.1 Expectation

The expectation of function is

$$\begin{aligned}\mathbb{E}[F(y(\xi), \cdot)] &= \int_{\mathcal{Y}} \rho^y(y) F(y(\xi), \cdot)\, dy = \Big\langle 1(y(\xi)) \;\Big|\; F(y(\xi), \cdot) \Big\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ &\approx \sum_{i=1}^{M} \alpha_i^y \times 1(y_i) \times F(y_i(\xi), \cdot).\end{aligned} \tag{7.2}$$

The derivative of expectation of function with respect to design variables $\xi$ is

$$\begin{aligned}\frac{d\mathbb{E}[F(y(\xi), \cdot)]}{d\xi} &= \frac{d}{d\xi}\left(\int_{\mathcal{Y}} \rho^y(y) F(y(\xi), \cdot)\, dy\right) = \int_{\mathcal{Y}} \frac{\partial \left(\rho^y(y) F(y(\xi), \cdot)\right)}{\partial \xi}\, dy \\ &= \int_{\mathcal{Y}} \rho^y(y) \frac{\partial F(y(\xi), \cdot)}{\partial \xi}\, dy \\ &= \mathbb{E}\left[\frac{\partial F(y(\xi), \cdot)}{\partial \xi}\right] \\ &\approx \sum_{i=1}^{M} \alpha_i^y \times 1(y_i) \times \frac{\partial F(y_i(\xi), \cdot)}{\partial \xi}\end{aligned} \tag{7.3}$$

### 7.1.2 Variance

The variance of function is

$$\begin{aligned}\mathbb{V}[F(y(\xi), \cdot)] &= \mathbb{E}[F(y(\xi), \cdot) \times F(y(\xi), \cdot)] &&- \quad \mathbb{E}[F(y(\xi), \cdot)] \times \mathbb{E}[F(y(\xi), \cdot)] \\ &= \Big\langle 1(y(\xi)) \;\Big|\; F^2(y(\xi), \cdot) \Big\rangle_{\rho^y(y)}^{\mathcal{Y}} &&- \quad \left(\Big\langle 1(y(\xi)) \;\Big|\; F(y(\xi), \cdot) \Big\rangle_{\rho^y(y)}^{\mathcal{Y}}\right)^2 \\ &\approx \sum_{i=1}^{M} \alpha_i^y \times F^2(y_i(\xi), \cdot) &&- \quad \left(\sum_{i=1}^{M} \alpha_i^y \times F(y_i(\xi), \cdot)\right)^2\end{aligned} \tag{7.4}$$

The derivative of variance is

$$\begin{aligned}
\frac{d\mathbb{V}[F(y(\xi),\cdot)]}{d\xi} &= \mathbb{E}\left[2F(y(\xi),\cdot) \times \frac{\partial F(y(\xi),\cdot)}{\partial \xi}\right] - 2\mathbb{E}[F(y(\xi),\cdot)] \times \frac{\partial \mathbb{E}[F(y(\xi),\cdot)]}{\partial \xi} \\
&= \mathbb{E}\left[2F(y(\xi),\cdot) \times \frac{\partial F(y(\xi),\cdot)}{\partial \xi}\right] - 2\mathbb{E}[F(y(\xi),\cdot)] \times \mathbb{E}\left[\frac{\partial F(y(\xi),\cdot)}{\partial \xi}\right] \\
&\approx \left(\sum_{i=1}^{M} \alpha_i^y \times 2F(y_i(\xi),\cdot) \times \frac{\partial F(y_i(\xi),\cdot)}{\partial \xi}\right) - 2\left(\sum_{i=1}^{M} \alpha_i^y \times F(y_i(\xi),\cdot)\right) \\
&\qquad \times \left(\sum_{i=1}^{M} \alpha_i^y \times \frac{\partial F(y_i(\xi),\cdot)}{\partial \xi}\right)
\end{aligned} \tag{7.5}$$

### 7.1.3 Standard Deviation

The standard deviation of function is

$$\mathbb{S}[F(y(\xi),\cdot)] = \sqrt{\mathbb{V}[F(y(\xi),\cdot)} \tag{7.6}$$

The derivative of standard deviation is

$$\frac{d\mathbb{S}[F(y(\xi),\cdot)]}{d\xi} = \frac{1}{2\sqrt{\mathbb{V}[F(y(\xi),\cdot)]}} \times \frac{\partial \mathbb{V}[F(y(\xi),\cdot)]}{\partial \xi} \tag{7.7}$$

The standard deviation and its derivative need no new additional evaluations. Note that all three probabilistic moments and its derivatives are obtained by repeatedly evaluating: $F(y(\xi),\cdot)$, $F^2(y(\xi),\cdot)$ and $\frac{\partial F(y(\xi),\cdot)}{\partial \xi}$. No modifications are necessary to existing source code that computes $F$ and $\frac{\partial F}{\partial \xi}$; therefore, this method is referred to as nonintrusive. Figure 7.1 illustrates this process; the red band indicates the nonlinear solution process at each node and the green band denotes the summation involved in the computation of quantities of interest.

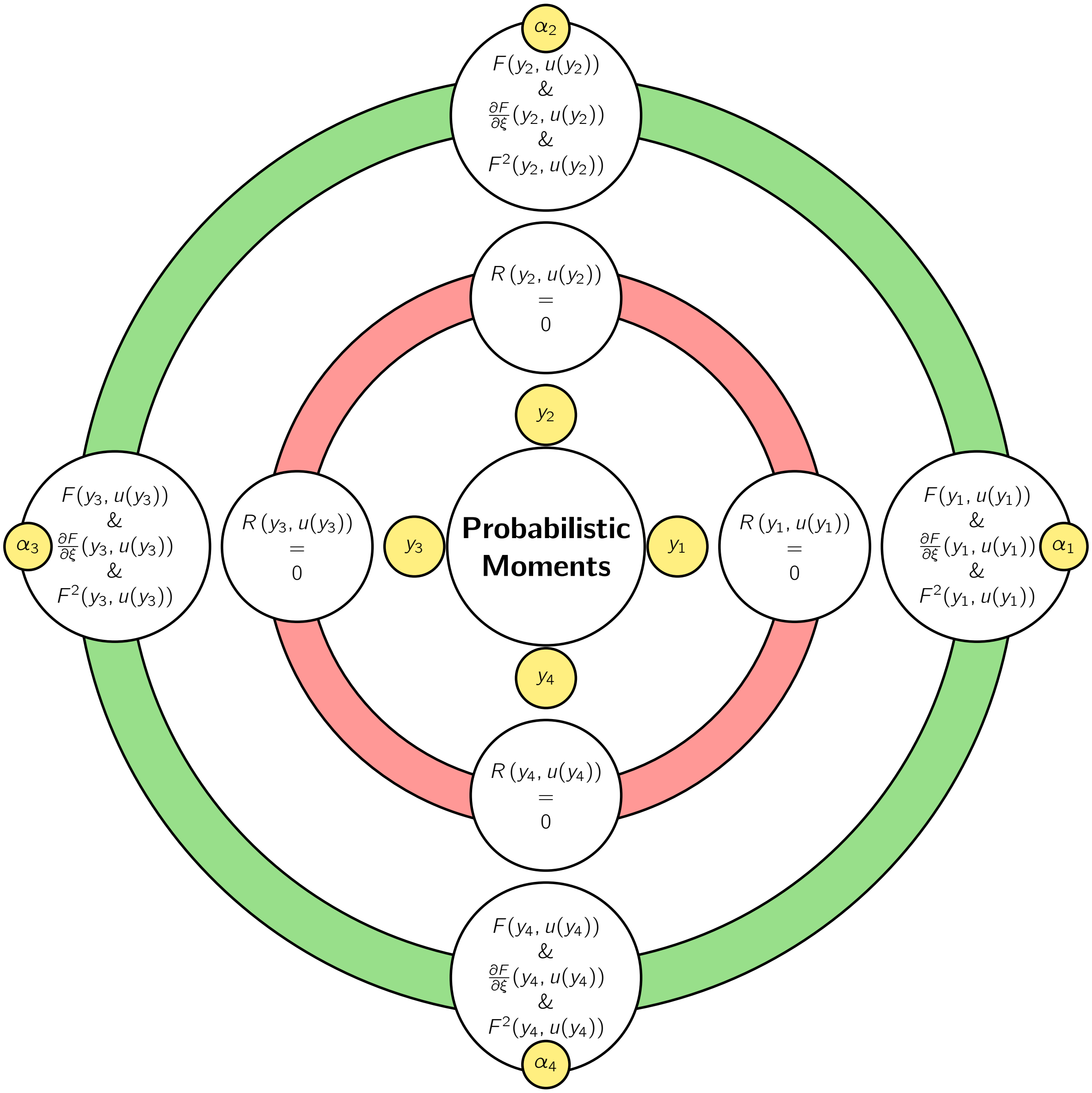

$\alpha_2$
$F(y_2,u(y_2))$ & $\frac{\partial F}{\partial \xi}(y_2,u(y_2))$ & $F^2(y_2,u(y_2))$
$R(y_2,u(y_2)) = 0$
$y_2$
$F(y_3,u(y_3))$ & $\frac{\partial F}{\partial \xi}(y_3,u(y_3))$ & $F^2(y_3,u(y_3))$
$\alpha_3$
$R(y_3,u(y_3)) = 0$
$y_3$
Probabilistic Moments
$y_1$
$R(y_1,u(y_1)) = 0$
$F(y_1,u(y_1))$ & $\frac{\partial F}{\partial \xi}(y_1,u(y_1))$ & $F^2(y_1,u(y_1))$
$\alpha_1$
$y_4$
$R(y_4,u(y_4)) = 0$
$F(y_4,u(y_4))$ & $\frac{\partial F}{\partial \xi}(y_4,u(y_4))$ & $F^2(y_4,u(y_4))$
$\alpha_4$

*Figure 7.1: Graphical illustration of stochastic sampling with four quadrature nodes.*

## 7.2 Semi-Intrusive Stochastic Galerkin Projection Method

In this section, we present the mathematical details of a semi-intrusive method of performing Galerkin projection in probabilistic domain to solve OUU problems governed by stochastic partial differential equations. We apply the finite element method for spatial discretization, implicit time marching methods for temporal discretization and discrete-adjoint method for sensitivity analysis [119, 120]. Our goal is to devise a strategy that parallels the solution process of deterministic problems and extending it into a stochastic analysis framework. This will enable the programming implementations to reuse the deterministic capabilities for spatial and temporal discretizations, while addressing the additional probabilistic domain.

### 7.2.1 Probabilistic Parameters, Basis and Quadrature

The creation of probabilistic parameters, setup of multivariate basis functions and quadrature rules used in this work are summarized next (see Chapter 6 for details).

**(a) Probabilistic Parameters:** We begin with a deterministic framework for PDEs where we specify coefficients for physical properties, initial and boundary conditions, as well as forcing functions. In order to incorporate the effect of uncertainties, we model the PDE inputs as variables that follow certain probability distribution types (see Table 7.1). This leaves us with a vector of random variables $y = [y_1, y_2, \ldots, y_M]$, where $M$ is the total number of probabilistically modeled random variables in the problem. Note that the random variables do not take the role of PDE inputs, but instead the PDE inputs are functions of random variables – the PDE inputs emerge from probabilistic parameter space.

*Table 7.1: Probability distributions, their standardized forms and orthonormal polynomials.*

| Distribution | PDF | Standardization | Standard PDF | Orthonormal Polynomials |
|---|---|---|---|---|
| Gaussian | $\frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right]$ | $z = \frac{y-\mu}{\sigma}$ | $\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}z^2\right)$ | $\frac{1}{0!}, \frac{z}{1!}, \frac{z^2-1}{2!}$<br>(Hermite) |
| Uniform | $\frac{1}{b-a}$ | $z = \frac{y-a}{b-a}$ | $1$ | $\sqrt{1}, (2z-1)\sqrt{3}, (6z^2-6z+1)\sqrt{5}$<br>(Legendre) |
| Exponential | $\frac{1}{\beta} \exp\left[-\left(\frac{y-\mu}{\beta}\right)\right]$ | $z = \frac{y-\mu}{\beta}$ | $\exp(-z)$ | $\frac{1}{0!}, \frac{-z+1}{1!}, \frac{z^2-4z+2}{2!}$<br>(Laguerre) |

**(b) Probabilistic Basis:** We select a set of orthonormal polynomials from the sequence of polynomials listed in Table 7.1, to serve as univariate basis functions along each probabilistic dimension based on the distribution type. The simplest option is to limit the highest allowed degree for each probabilistic variable. An univariate basis can be extended to multivariate basis using tensor product rule, complete polynomial rule or others. Finally, we obtain the following orthonormal multivariate basis set

$$\widehat{\psi}(y) = \{\widehat{\psi}_1(y), \widehat{\psi}_2(y), \ldots, \widehat{\psi}_N(y)\}, \tag{7.8}$$

where $N$ is the total number of multivariate basis functions that act as stochastic degrees of freedom that are associated with each deterministic spatio-temporal degrees of freedom. As a consequence of this property, a trivial unicardinal basis set $\widehat{\psi}(y) = \{1\}$ simply recovers the deterministic problem, which is handy during the verification and validation of stochastic Galerkin framework. The orthonormality of any two basis functions $\widehat{\psi}_i(y)$ and $\widehat{\psi}_j(y)$ is mathematically defined as

$$\left\langle \widehat{\psi}_i(y) \;\middle|\; \widehat{\psi}_j(y) \right\rangle^{\mathcal{Y}}_{\rho^y(y)} = \int_{\mathcal{Y}} \widehat{\psi}_i(y) \rho^y(y) \widehat{\psi}_j(y) \; dy = \begin{cases} 1 & \text{if } \; i = j \\ 0 & \text{if } \; i \neq j \end{cases}, \tag{7.9}$$

where $\rho^y(y) \geq 0$ for $y \in \mathcal{Y}$ is the probability density function. Since, the orthonormal polynomial set is derived based on the standard random variable $z \in \mathcal{Z}$ (see Table 7.1), the calculus rules for transformation of variables must be used when evaluating inner products in the physical probabilistic domain $\mathcal{Y}$. Note that the standardization is usually done with zero location and unit stretch as distribution parameters.

**(c) Probabilistic Quadrature:** We obtain one dimensional quadrature points based on the distribution type and perform tensor product along each dimension to setup multivariate quadrature to approximate integrals arising in projection. We refer to the resulting set of weights and quadrature points as $\{\alpha_q, y_q\}_{q=1}^{Q}$, where $\alpha_q$ refers to the $q - th$ scalar weight, $y_q$ refers to the $q - th$ quadrature point vector, and $Q$ is the total number of quadrature points. Instead of full tensor product, the sparse quadrature methods can also be used when appropriate smoothness justifications can be made; but these are not explored in this work.

### 7.2.2 Temporal Physical Analysis

The residual of the stochastic governing equations can be written in implicit form as

$$\begin{aligned} R\left(t, y, u(t,y), \dot{u}(t,y), \ddot{u}(t,y)\right) &= 0 \\ u(0,y) &= u_0(y) \\ \dot{u}(0,y) &= \dot{u}_0(y) \end{aligned} \tag{7.10}$$

where $u(t,y)$ are the functions characterizing the physical state of the stochastic system, with their corresponding first and second time derivatives, $\dot{u}(t,y)$ and $\ddot{u}(t,y)$, respectively. Note that $t$ is the temporal variable from time domain $\mathcal{T}$, and $y$ is the random variable from probabilistic domain $\mathcal{Y}$.

#### *7.2.2.1 Formation of Stochastic States*

We begin the solution process to (7.10) with the following hypothesis for stochastic state functions:

$$\begin{aligned} u(t,y) &\approx \sum_{i=1}^{N} u_i(t)\widehat{\psi}_i(y) \\ \dot{u}(t,y) &\approx \sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i(y) \\ \ddot{u}(t,y) &\approx \sum_{i=1}^{N} \ddot{u}_i(t)\widehat{\psi}_i(y) \end{aligned} \tag{7.11}$$

Eq. (7.11) applies the principle of *superposition* and *separation of variables* for solving differential equations. Due to global support of the basis functions $\widehat{\psi}(y)$ used in this work, these can be referred to as spectral expansion of stochastic fields (polynomial chaos expansions).

*7.2.2.2 Formation of Stochastic Residual*

The stochastic residual vector can be computed using repeated evaluations of the deterministic residual vector for each quadrature point. The $i$-th stochastic residual component is the coefficient resulting from the projection of (7.10) on the $i$-th basis set component $\widehat{\psi}_i(y)$:

$$\begin{aligned}\mathcal{R}_i &\triangleq \left\langle \widehat{\psi}_i(y) \;\middle|\; R(t, y, u(t,y), \dot{u}(t,y), \ddot{u}(t,y)) \right\rangle^{\mathcal{Y}}_{\rho^y(y)} \\ &\approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_i(y_q)}_{\text{scalar}} \underbrace{R(t, y_q, u(t,y_q), \dot{u}(t,y_q), \ddot{u}(t,y_q))}_{\text{deterministic residual evaluated at } y_q}.\end{aligned} \tag{7.12}$$

We use numerical quadrature to approximate this inner product with $Q$ quadrature points from the probabilistic space $\mathcal{Y}$. The number of quadrature points necessary can sometimes be determined *a priori* from the polynomial degree of the integrand, and can be used to optimize computations. The full stochastic residual vector takes the form:

$$\mathcal{R} = \begin{bmatrix} \mathcal{R}_1 \\ \mathcal{R}_2 \\ \vdots \\ \mathcal{R}_N \end{bmatrix} = \begin{bmatrix} \sum\limits_{q=1}^{Q} \alpha_q \widehat{\psi}_1(y_q) R(t, y_q, u(t,y_q), \dot{u}(t,y_q), \ddot{u}(t,y_q)) \\ \sum\limits_{q=1}^{Q} \alpha_q \widehat{\psi}_2(y_q) R(t, y_q, u(t,y_q), \dot{u}(t,y_q), \ddot{u}(t,y_q)) \\ \vdots \\ \sum\limits_{q=1}^{Q} \alpha_q \widehat{\psi}_N(y_q) R(t, y_q, u(t,y_q), \dot{u}(t,y_q), \ddot{u}(t,y_q)) \end{bmatrix} \tag{7.13}$$

The Newton-Raphson iterative process for nonlinear solution [119, 120] of governing equations (7.10) uses assumed stochastic state values $\mathbb{U}(t) = [u_1(t), u_2(t), \ldots, u_N(t)]$, based on which $u(t, y_q)$ are evaluated following (7.11). The size of the stochastic residual vector is $N$ times the size of deterministic residual vector, as a direct consequence of the setup of the probabilistic basis functions. In physics-based simulations, one works with element-wise residuals and assembled system-wide residuals; the approach that we present here applies to system-wide *and* element-wise residuals. Based on Eqs. (7.12) and (7.13), we show that the stochastic residual can be computed implicitly using repeated evaluations of the deterministic residual at predetermined parameter values $y_q$. When one applies sampling-based approach for uncertainty propagation, the deterministic code is evaluated for samples points $y_q$ based on which statistics of the output metrics are computed as discussed in Section 7.1, referring to the class of methods as “non-intrusive”. We refer to our approach as “semi”-intrusive, because the deterministic simulation codes should have the flexibility to update parameters so as to recompute residuals, which may or may not have been be implemented *a priori* based on the software architecture.

#### *7.2.2.3 Formation of Stochastic Jacobian*

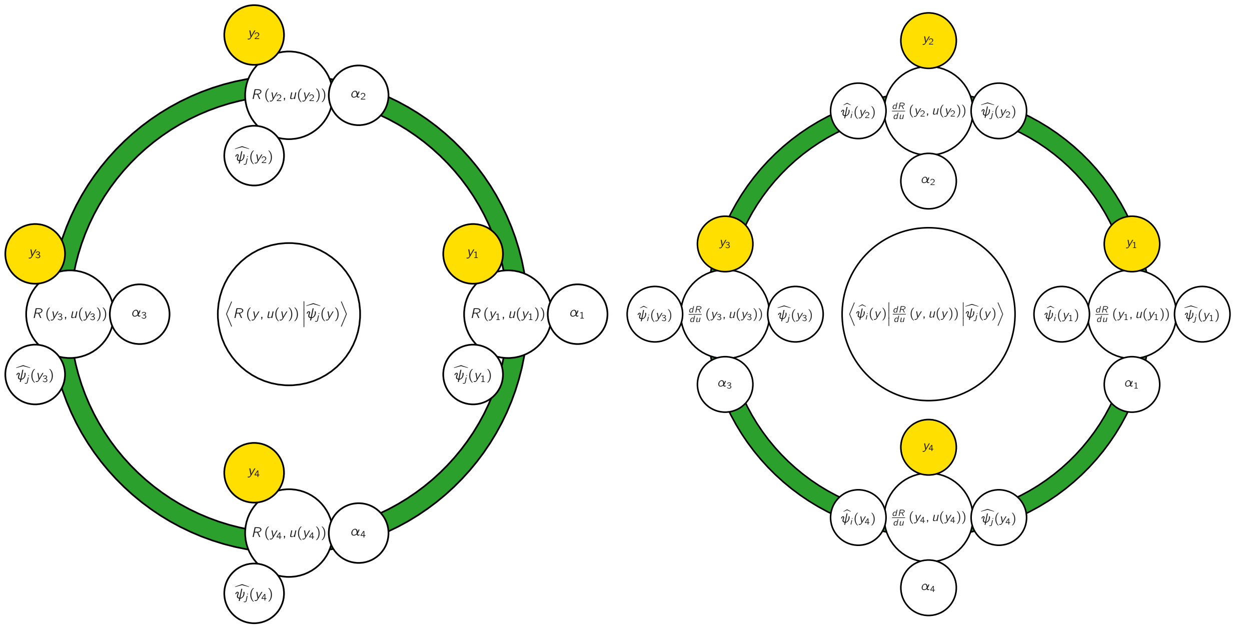
The stochastic Jacobian matrix (the Jacobian matrix of the stochastic ODE/DAE) can be computed using repeated evaluations of the deterministic Jacobian (the Jacobian matrix of the deterministic ODE/DAE) for each quadrature point. The block component of the stochastic Jacobian is

$$\begin{aligned} \mathcal{J}_{i,j} &\triangleq \left\langle \widehat{\psi}_i(y) \,\middle|\, J(t, y, u(t, y), \dot{u}(t, y), \ddot{u}(t, y)) \,\middle|\, \widehat{\psi_j}(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ &\approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_i(y_q) \widehat{\psi}_j(y_q)}_{\text{scalar}} \underbrace{J(t, y_q, u(t, y_q), \dot{u}(t, y_q), \ddot{u}(t, y_q))}_{\text{deterministic Jacobian evaluated at } y_q} \end{aligned} \tag{7.14}$$

where $J$ is the deterministic Jacobian and $Q$ is the number of quadrature points. The full stochastic Jacobian matrix takes the form

$$\mathcal{J} = \begin{bmatrix} \mathcal{J}_{1,\,1} & \mathcal{J}_{1,\,2} & \dots & \mathcal{J}_{1,\,N} \\ \mathcal{J}_{2\,1} & \mathcal{J}_{2,\,2} & \dots & \mathcal{J}_{2,\,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{J}_{N,\,1} & \mathcal{J}_{N,\,2} & \dots & \mathcal{J}_{N,\,N} \end{bmatrix}. \tag{7.15}$$

**A graphical illustration and scope for parallelism:** The formation of stochastic residual and Jacobian through projection is illustrated in Figure 7.2. As it can be seen, the required items are the basis functions, quadrature nodes with weights and deterministic residual/Jacobian counterparts. The green band refers to the summation involved in quadrature approximation of inner products. Depending on the implementation of the underlying deterministic code, the formation of stochastic residuals and Jacobians can be done in parallel.


$y_2$
$R\left(y_2, u(y_2)\right)$
$\alpha_2$
$\widehat{\psi_j}(y_2)$
$y_3$
$R\left(y_3, u(y_3)\right)$
$\alpha_3$
$\widehat{\psi_j}(y_3)$
$\left\langle R\left(y, u(y)\right) \middle| \widehat{\psi_j}(y) \right\rangle$
$y_1$
$R\left(y_1, u(y_1)\right)$
$\alpha_1$
$\widehat{\psi_j}(y_1)$
$y_4$
$R\left(y_4, u(y_4)\right)$
$\alpha_4$
$\widehat{\psi_j}(y_4)$
$y_2$
$\widehat{\psi_i}(y_2)$
$\frac{dR}{du}\left(y_2, u(y_2)\right)$
$\widehat{\psi_j}(y_2)$
$\alpha_2$
$y_3$
$\widehat{\psi_i}(y_3)$
$\frac{dR}{du}\left(y_3, u(y_3)\right)$
$\widehat{\psi_j}(y_3)$
$\alpha_3$
$\left\langle \widehat{\psi_i}(y) \middle| \frac{dR}{du}\left(y, u(y)\right) \middle| \widehat{\psi_j}(y) \right\rangle$
$y_1$
$\widehat{\psi_i}(y_1)$
$\frac{dR}{du}\left(y_1, u(y_1)\right)$
$\widehat{\psi_j}(y_1)$
$\alpha_1$
$y_4$
$\widehat{\psi_i}(y_4)$
$\frac{dR}{du}\left(y_4, u(y_4)\right)$
$\widehat{\psi_j}(y_4)$
$\alpha_4$

*(a) Projection of residual*

*(b) Projection of Jacobian*

*Figure 7.2: The formation of stochastic residual and Jacobian entries using inner products approximated using quadrature.*

**Sparsity and symmetry:** The question of the sparsity and symmetry of the stochastic Jacobian matrix arises naturally in the interest of efficient computations. Since we use normalized basis functions for projection (orthonormal polynomials), we have a symmetry stochastic Jacobian. The nonzero entries (sparse entries) of the Jacobian can be determined, if the polynomial degree of the probabilistically modeled PDE inputs is known. This can offer significant computational savings and pave way for matrix-free implementations that are suitable for parallel scaling. The sparsity is also dependent on the choice of basis functions, and their ordering. The study of sparsity of stochastic Galerkin matrices is reported in Ernst and Ullmann [81]. In Section 7.2.4, we perform a study on the sparsity patterns resulting from basis selection using tensor-product rule and basis selection using complete polynomial rule for different stateless Jacobians.

#### *7.2.2.4 Initial Conditions*

The initial conditions of the stochastic ODE can be formed by reevaluating the deterministic initial conditions for each quadrature point. The i-th component of the stochastic initial condition vector is formed as

$$\left\langle \widehat{\psi}_i(y) \,\middle|\, u_0(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_i(y_q)}_{\text{scalar}} \quad \underbrace{u_0(y_q)}_{\text{deterministic ICs for } y_q} \tag{7.16}$$

The treatment of boundary conditions of the PDE follow in a similar fashion.

### 7.2.3 Adjoint Sensitivity Analysis

For solving the probabilistic OUU problem (2.17), we require the derivatives of the expectation, standard deviation and variance of the metrics of interest, with respect to the design variables. The adjoint sensitivity analysis involves solving the adjoint equations and forming the total derivative based on the adjoint variables. We detail the extension of deterministic adjoint formulation to stochastic adjoint by reusing the deterministic capabilities. The deterministic adjoint formulations used here are published in Boopathy and Kennedy [119, 120].

#### *7.2.3.1 Expectation Operator*

The expectation of metric of interest $F(t, y, u(t, y), \dot{u}(t, y), \ddot{u}(t, y)))$ denoted as $F(y, \cdot)$ is

$$\mathbb{E}[F(y,\cdot)] = \left\langle \widehat{\psi}_1(y) \,\middle|\, F(y,\cdot) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \; \underbrace{F(y_q,\cdot)}_{\text{deterministic}} \tag{7.17}$$

The deterministic metric is evaluated repeatedly at points corresponding to probabilistic quadrature to obtain the expectation operator. The derivative of expectation of the metric with respect to the design variables follows as

$$\frac{d\mathbb{E}\left[F(y,\cdot)\right]}{d\xi} = \left\langle \widehat{\psi}_1(y) \;\middle|\; \frac{dF(y,\cdot)}{d\xi} \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{\frac{dF(y_q,\cdot)}{d\xi}}_{\text{deterministic}} . \tag{7.18}$$

The deterministic adjoint derivative of $F(y_q, \cdot)$ is evaluated as

$$\frac{dF(y_q,\cdot)}{d\xi} = \frac{\partial F(y_q,\cdot)}{\partial \xi} + \lambda^T(t, y_q)\frac{\partial R(y_q,\cdot)}{\partial \xi}, \tag{7.19}$$

where the stochastic adjoint states

$$\lambda(t,y) \approx \sum_{i=1}^{N} \lambda_i(t)\widehat{\psi}_i(y) \tag{7.20}$$

are solved from the linear system

$$\begin{bmatrix} \mathcal{J}_{1,1} & \mathcal{J}_{1,2} & \dots & \mathcal{J}_{1,N} \\ \mathcal{J}_{2\,1} & \mathcal{J}_{2,2} & \dots & \mathcal{J}_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{J}_{N,1} & \mathcal{J}_{N,2} & \dots & \mathcal{J}_{N,N} \end{bmatrix}^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix} = - \begin{bmatrix} \left(\partial F(\cdot)/\partial u\right)_1 \\ \left(\partial F(\cdot)/\partial u\right)_2 \\ \vdots \\ \left(\partial F(\cdot)/\partial u\right)_N \end{bmatrix} \tag{7.21}$$

The right hand side terms are formed by projecting deterministic terms as

$$\begin{aligned} \left(\partial F/\partial u\right)_i &= \left\langle \widehat{\psi}_1(y) \;\middle|\; \partial F(y,\cdot)/\partial u \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ &\approx \underbrace{\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{\partial F(y_q,\cdot)/\partial u}_{\text{determinisic rhs at } y_q} \end{aligned} \tag{7.22}$$

In time dependent adjoint formulations, the right hand side terms are composed of contributions that are not solely the partial derivatives of metrics $\partial F/\partial u$ (see Chapter 4). In this scenario, the fully formed right hand side can be projected as outlined in (7.21).

#### *7.2.3.2 Variance Operator*

The variance is

$$\begin{aligned}
\mathbb{V}[F(y,\cdot)] &= \mathbb{E}[F^2(y,\cdot)] - \mathbb{E}[F(y,\cdot)]^2 \\
&= \left\langle \widehat{\psi}_1(y) \;\middle|\; F^2(y,\cdot) \right\rangle^{\mathcal{Y}}_{\rho^y(y)} - \left\langle \widehat{\psi}_1(y) \;\middle|\; F(y,\cdot) \right\rangle^{\mathcal{Y}\quad 2}_{\rho^y(y)} \\
&\approx \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{F^2(y_q,\cdot)}_{\text{deterministic}} - \left[ \sum_{q=1}^{Q} \underbrace{\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{F(y_q,\cdot)}_{\text{deterministic}} \right]^2
\end{aligned} \tag{7.23}$$

The derivative of variance follows as

$$\frac{d\mathbb{V}[F(y,\cdot)]}{d\xi} = \frac{d\mathbb{E}[F^2(y,\cdot)]}{d\xi} - 2\mathbb{E}[F(y,\cdot)]\frac{d\mathbb{E}[F(y,\cdot)]}{d\xi} \tag{7.24}$$

where the first term

$$\begin{aligned}
\frac{d\mathbb{E}[F^2(y,\cdot)]}{d\xi} &= \left\langle \widehat{\psi}_1(y) \;\middle|\; 2F(y,\cdot)\frac{dF(y,\cdot)}{d\xi} \right\rangle^{\mathcal{Y}}_{\rho^y(y)} \\
&\approx \sum_{q=1}^{Q} \underbrace{2\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{F(y_q,\cdot)\frac{dF(y_q,\cdot)}{d\xi}}_{\text{deterministic}}
\end{aligned} \tag{7.25}$$

The deterministic adjoint derivative of $F^2(y_q,\cdot)$ is evaluated as

$$\frac{dF^2(y_q,\cdot)}{d\xi} = \frac{\partial F^2(y_q,\cdot)}{\partial \xi} + \phi^T(t,y_q)\frac{\partial R(y_q,\cdot)}{\partial \xi}. \tag{7.26}$$

The stochastic adjoint states

$$\phi(t,y) \approx \sum_{i=1}^{N} \phi_i(t)\widehat{\psi}_i(y) \tag{7.27}$$

are solved from the linear system

$$\begin{bmatrix} \mathcal{J}_{1,1} & \mathcal{J}_{1,2} & \dots & \mathcal{J}_{1,N} \\ \mathcal{J}_{2\,1} & \mathcal{J}_{2,2} & \dots & \mathcal{J}_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{J}_{N,1} & \mathcal{J}_{N,2} & \dots & \mathcal{J}_{N,N} \end{bmatrix}^T \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_N \end{bmatrix} = - \begin{bmatrix} \left(\partial F^2(\cdot)/\partial u\right)_1 \\ \left(\partial F^2(\cdot)/\partial u\right)_2 \\ \vdots \\ \left(\partial F^2(\cdot)/\partial u\right)_N \end{bmatrix} \tag{7.28}$$

The right hand side terms are formed by projecting deterministic terms as

$$\begin{aligned} \left(\frac{\partial F^2(y,\cdot)}{\partial u}\right)_i &= \left\langle \widehat{\psi}_1(y) \;\middle|\; 2F(y,\cdot)\frac{\partial F(y,\cdot)}{\partial u} \right\rangle^{\mathcal{Y}}_{\rho^y(y)} \\ &\approx \underbrace{2\alpha_q \widehat{\psi}_1(y_q)}_{\text{scalar}} \underbrace{F(y_q,\cdot)\frac{\partial F(y_q,\cdot)}{\partial u}}_{\text{determinisic rhs at } y_q} \end{aligned} \tag{7.29}$$

#### *7.2.3.3 Standard Deviation Operator*

The standard deviation of a function of interest can be obtained from the variance

$$\mathbb{S}[F(y,\cdot)] = \sqrt{\mathbb{V}[F(y,\cdot)}. \tag{7.30}$$

The derivative of standard deviation is

$$\frac{d\mathbb{S}[F(y,\cdot)]}{d\xi} = \frac{1}{2\sqrt{\mathbb{V}[F(y,\cdot)]}} \cdot \frac{d\mathbb{V}[F(y,\cdot)]}{d\xi} \tag{7.31}$$

### 7.2.4 Study of Stochastic Projection Matrices

In this section, we are interested in studying the pattern of non-zero entries of the Jacobian matrices arising in stochastic computations. By knowing the nonzero patterns apriori or real-time allows one to efficiently perform stochastic numerical computations. The study of sparsity of stochastic Galerkin matrices is reported in works such as Ernst and Ullmann [81]. Apart from deducing conditions on when the coefficients are nonzero, we also study the numerical aspects such as the minimum possible number of quadrature points for optimal evaluation of inner products.

#### *7.2.4.1 Entries of the Jacobian Matrix*

Let $y = [y_1, y_2, \ldots, y_N]$ be a vector of random variables. If we decompose one of the basis entry, say $\psi_i(y)$, with respect to another basis entry, say $\psi_j(y)$, we get the decomposition coefficient as either one (when $i = j$) or zero (when $i \neq j$) owing to the orthonormality of basis functions by construction. However, if we project a function, say $h(y) = f(y) \times \psi_j(y)$, on one of the basis functions $\psi_i(y)$, we shall get a scalar as the decomposition coefficient, depending on the actual form of the function $f(y)$. We use the term Jacobian to refer to the two-dimensional arrangement of decomposition coefficients $\mathcal{J}$ (a second-order tensor) indexed by an ordered tuple $(i, j)$. Formally, the (i,j)-th entry of Jacobian matrix results from the inner product (projection) defined as

$$\mathcal{J}_{ij} = \Big\langle \psi_i(y) \;\Big|\; f(y) \;\Big|\; \psi_j(y) \Big\rangle_{\rho(y)}^{\mathcal{Y}} = \int_{\mathcal{Y}} \psi_i(y) f(y) \psi_j(y) \rho(y) dy \approx \sum_{q=1}^{Q} \alpha_q \psi_i(y_q) f(y_q) \psi_j(y_q). \tag{7.32}$$

Here $Q$ is the total number of multivariate quadrature points and $\alpha_q$ are the associated weights. The sparsity of Jacobian matrix $\mathcal{J}$ depends on the functional form of $f(y)$ as it can be a polynomial function like $y^2 - y$, or a trigonometric like $\sin(y)$, or a rational function like $1/(1+y^2)$. Although $f(y)$ may assume any functional form as mentioned above, for the type of partial differential equations of engineering relevance and of interest in this work, $f(y)$ takes simple polynomial forms and is a placeholder for coefficients of the partial differential equation.

#### *7.2.4.2 Degree of Integrand and Quadrature Points*

Consider a function of three variables $f(y_1, y_2, y_3)$ to assume forms listed in Table 7.2 with corresponding variable-wise degrees $[d_1, d_2, d_3]$. Recall that for polynomial function in more

*Table 7.2: Functional forms and corresponding variable-wise degrees.*

| No. | Functional form of $f(y)$ | Variable-wise degree |
|---|---|---|
| 1 | $y_1$ | $[1,0,0]$ |
| 2 | $y_1y_2$ | $[1,1,0]$ |
| 3 | $y_2^3$ | $[0,3,0]$ |
| 4 | $y_1^2y_2^2+3y_1^3y_3+4y_2$ | [3,2,1] |
| 5 | $y_1y_3+\sin(y_1)$ | $[\infty,0,1]$ |
| 6 | $y_1y_2+\cos(y_1,y_3)$ | $[\infty,1,\infty]$ |
| 7 | $y_1y_2+\exp(y_1,y_3)$ | $[\infty,1,\infty]$ |

than one variable, the degree of a term is the sum of exponents of the variables. The total degree of the polynomial is the maximum of the term-wise degrees. However, instead of total highest degree, we work with variable-wise highest degrees that facilitates constructing quadrature rules based on the actual degree of the variable thereby reducing the number of evaluations of $f(y)$. Recall that the number of quadrature points required to integrate a function of degree $d$ is $Q = 2(d) - 1$. Based on this rule, Table 7.3 lists the number of points and the polynomials that are exactly integrated.

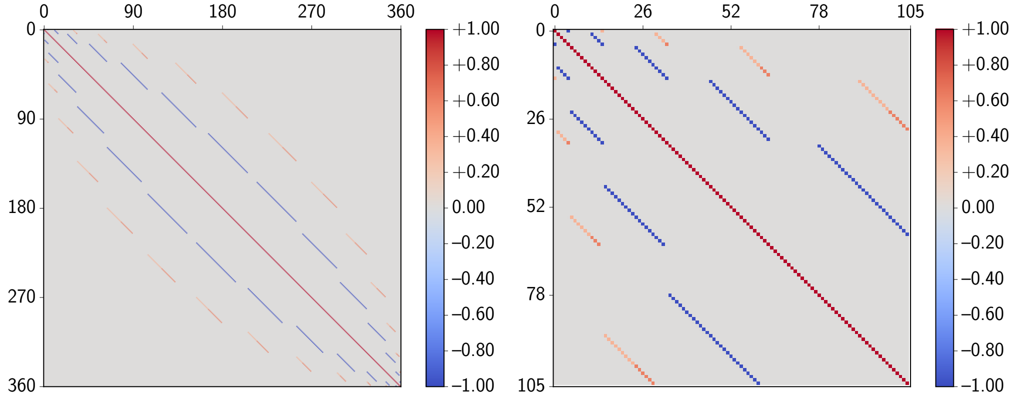
*Table 7.3: Degree of operand for different mathematical operations.*

| number of quadrature points | polynomials $P_i(y)$ |
|---|---|
| 1 | $P_0, P_1$ |
| 2 | $P_0, P_1, P_2, P_3$ |
| 3 | $P_0, P_1, P_2, P_3, P_4, P_5$ |

#### *7.2.4.3 Sparsity for Tensor and Complete Polynomial Spaces*

Let us analyze the sparsity patterns resulting from basis selection using tensor-product rule and basis selection using complete polynomial rule, on a variety of stateless Jacobian functions. The number of primary off-diagonal bands depend on the nonlinearity of the function $f(y)$.

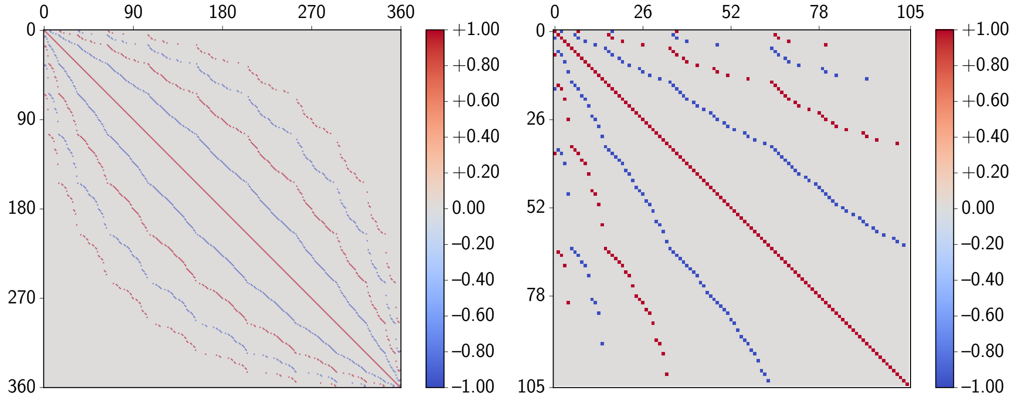
**Uncoupled Sparsity** From Figure 7.3 we see that there are two primary off-diagonal bands which in the exponent on the first-random variable.



*(a) tensor basis*

*(b) complete basis*

*Figure 7.3: Jacobian of decomposition for* $f(y) = y_1^2$ *for* $d_1 = 2, d_2 = 0, d_3 = 0, d_4 = 0$.



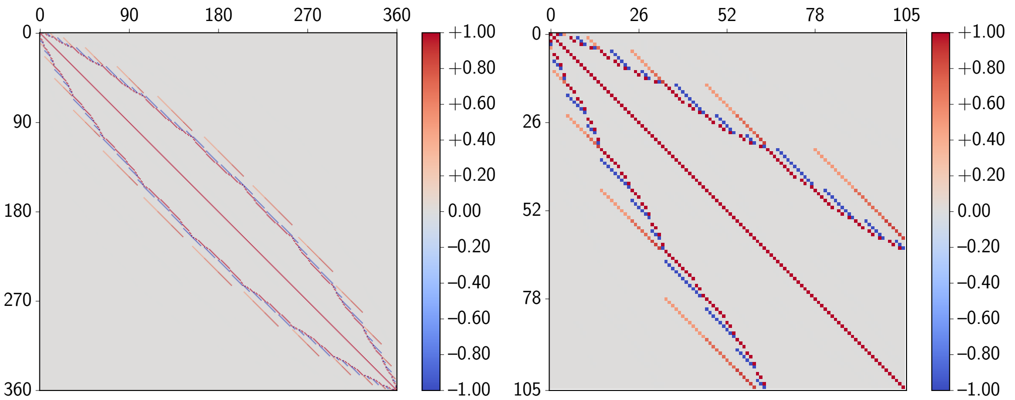
*(a) tensor basis*

*(b) complete basis*

*Figure 7.4: Jacobian of decomposition for* $f(y) = y_3^4$ *for* $d_1 = 0, d_2 = 0, d_3 = 4, d_4 = 0$.

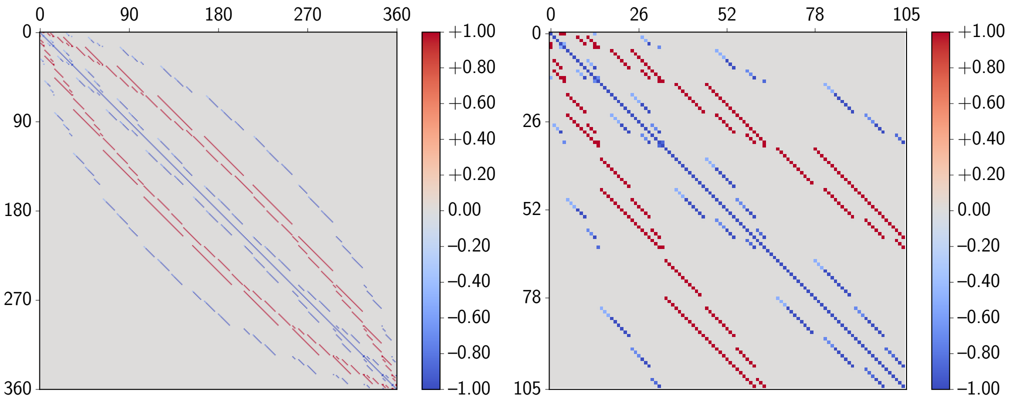
From Figure 7.4 we see that there are four primary off-diagonal bands which in the exponent on the third-random variable.

**Coupled Sparsity** When there is a coupling among probabilistic dimensions (variables), for example $y_1y_2$ (not $y_1+y_2$), we observe secondary bands. Figures 7.6, 7.7 and 7.8 illustrate the presence of secondary bands resulting from coupling between probabilistic dimensions.



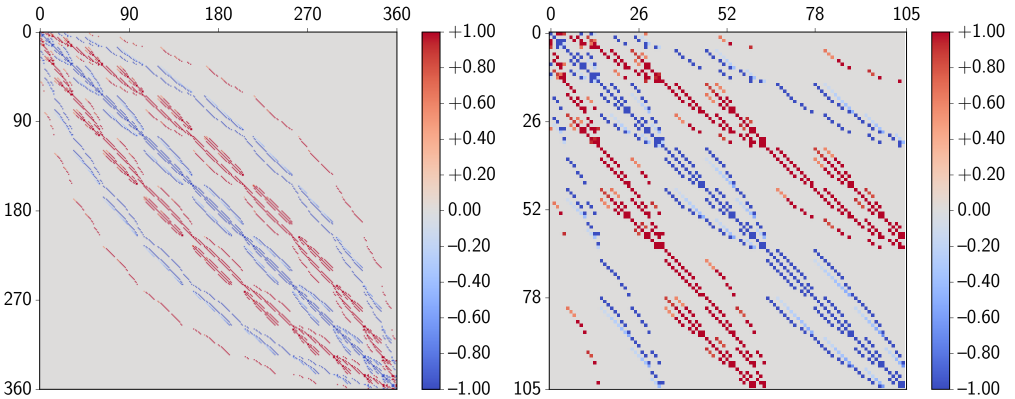
*(a) tensor basis* *(b) complete basis*

*Figure 7.5: Jacobian of decomposition for* $f(y) = y_1 + y_2 + y_3$ *for* $d_1 = 1, d_2 = 1, d_3 = 1, d_4 = 0$.



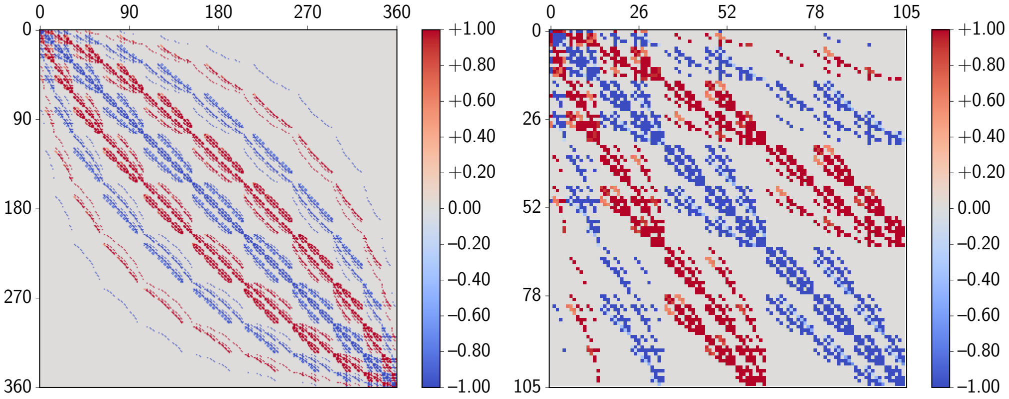
*(a) tensor basis* *(b) complete basis*

*Figure 7.6: Jacobian of decomposition for* $f(y) = y_1 y_2$ *for* $d_1 = 1, d_2 = 1, d_3 = 0, d_4 = 0$.



*(a) tensor basis* *(b) complete basis*

*Figure 7.7: Jacobian of decomposition for* $f(y) = y_1 y_2 y_3$ *for* $d_1 = 1, d_2 = 1, d_3 = 1, d_4 = 0$.



*(a) tensor basis* *(b) complete basis*

*Figure 7.8: Jacobian of decomposition for $f(y) = y_1 y_2 y_3 y_4$ for $d_1 = 1, d_2 = 1, d_3 = 1, d_4 = 1$.*

## 7.3 Implementation of Semi-Intrusive Stochastic Galerkin Method

We proceed with the subject of computer implementation of semi-intrusive SGM. We begin with the summary of the deterministic and stochastic finite element methodologies in Table 7.4. The mesh information (nodes and connectivities) are identical between the two methods, but differ in residual, Jacobians and metrics of interest.

*Table 7.4: Summary of deterministic and stochastic finite element methodologies.*

| | **Deterministic FEM** | **Stochastic FEM** |
|---|---|---|
| nodes | coordinates of position vectors of points discretizing the spatial geometry | coordinates of position vectors of points discretizing the spatial geometry |
| connectivity | ordered sequence of nodes define a deterministic element | the same sequence of nodes define a stochastic element |
| state (field) variables | spatial and temporal degrees of freedom | spatial, temporal, and probabilistic degrees of freedom |
| residuals, Jacobians | vectors and matrices that are derived from deterministic ODE | vectors and matrices that are $N$ times larger than deterministic ODEs |
| metrics of interest | integrals (or functions of integrals of) in space and time domains | moments of integrals (or functions of integrals of) in space and time domains |

### 7.3.1 Software Architecture for Element-wise Projection

Often, finite element simulation libraries use object-oriented programming principles such as abstraction, inheritance and composition. We outline the object-oriented software architecture that is used to develop the stochastic finite element framework by extending the deterministic finite element framework TACS [119, 120]. Figure 7.9 shows the core element-level software architecture followed in this work. The deterministic finite element framework for multibody dynamics has a pool of concrete classes implementing the `Element` interface. We refer to these concrete classes with generic names `Element Type 1` and `Element Type 2`, but these are often beams, shells, solids, rigid bodies, kinematic constraints in the context of multibody dynamics. These concrete classes provide us the means to obtain deterministic residuals, Jacobian matrices, as well as derivatives needed for adjoint sensitivity analysis. This forms the core of deterministic adjoint-enabled finite element framework. In order to extend this framework to implement stochastic Galerkin finite element method, we provide two new classes `ProbabilisticSpace` and `StochasticElement`, that function as follows:

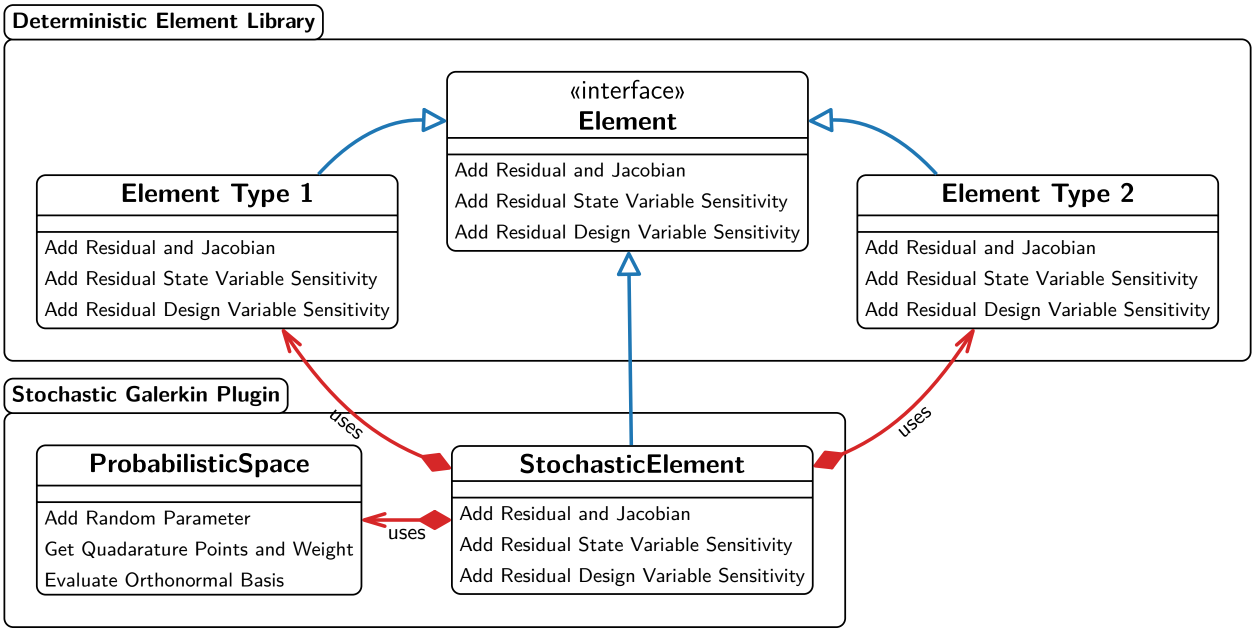

Deterministic Element Library
«interface»
Element
Add Residual and Jacobian
Add Residual State Variable Sensitivity
Add Residual Design Variable Sensitivity
Element Type 1
Add Residual and Jacobian
Add Residual State Variable Sensitivity
Add Residual Design Variable Sensitivity
Element Type 2
Add Residual and Jacobian
Add Residual State Variable Sensitivity
Add Residual Design Variable Sensitivity
Stochastic Galerkin Plugin
uses
uses
ProbabilisticSpace
Add Random Parameter
Get Quadarature Points and Weight
Evaluate Orthonormal Basis
uses
StochasticElement
Add Residual and Jacobian
Add Residual State Variable Sensitivity
Add Residual Design Variable Sensitivity

*Figure 7.9: The element-level software architecture of semi-intrusive stochastic Galerkin method for projection in probabilistic space.*

1. The `ProbabilisticSpace` class acts as a container for probabilistically modeled random parameters, as well as is responsible for multivariate basis evaluation and quadrature setup. This class abstracts the different options for basis function evaluation and quadrature rules.

2. The `StochasticElement` class implements the `Element` interface and uses one of the concrete deterministic element instances. Thus, the `StochasticElement` is a regular element through *inheritance*, and also has a deterministic element through *composition*. The composition enables the reuse of underlying deterministic capabilities for forward analysis and adjoint sensitivity analysis. The inheritance allows us to utilize the existing assembly, linear algebra and time-marching algorithms.

In addition to the architecture of the element library shown in Figure 7.9, there are other classes such as the `Assembler` and the `TimeIntegrator` responsible for system-wide assembly operations, and implicit time marching, respectively (see Section 4.6 for details). The evaluation of metrics of interest can be accomplished through a similar setup of `Function` – `StochasticFunction` using `ProbabilisticSpace`. The evaluation of deterministic metrics such as stress, compliance and structural mass, happens through specialized implementations of the `Function` interface. The `StochasticFunction` is responsible for computing the probabilistic moments of such deterministic metrics.

**System-wide and Element-wise Projection:** We presented the architecture for element-wise stochastic Galerkin projection. As mentioned earlier in this section, it is also possible to perform Galerkin projection at the system level operating on larger system wide matrices and vectors. In order to achieve that, the architecture would involve the same `ProbabilisticSpace` class operating on `StochasticAssembler` extending `Assembler` interface. Both options are mathematically equivalent, and thus can be decided based on the convenience of implementation, and software architecture of the deterministic code base.

### 7.3.2 Software Algorithms for Element-wise Projection

We present basic computational algorithms forming the core of nonlinear stochastic solution process. A nonlinear solution process is driven by assembling residuals, Jacobians and initial conditions (e.g. Newton–Raphson method). The goal here is to present the key mathematical operations in the form of algorithms, thus the presented algorithms need not be an optimal implementation for fast computations.

#### 7.3.2.1 *Stochastic Element Residual*

Let the deterministic residual of an element $e$ be $R^e\left(t, y, u^e(t,y), \dot{u}^e(t,y), \ddot{u}^e(t,y)\right)$ and the stochastic residual of element $e$ be $\mathcal{R}^e\left(t, y, U^e(t), \dot{U}^e(t), \ddot{U}^e(t)\right)$. The size of stochastic element residual vector is $N$ times the size of deterministic element residual vector, where the integer multiple is the number of basis functions in the orthonormal set. For every orthonormal basis function $i = 1, \dots, N$, we project the element residual on basis function $\widehat{\psi}_i^y(y)$ and place the decomposition coefficients in the stochastic element residual array. This projection is evaluated using numerical quadrature as

$$\begin{gathered}\left\langle \widehat{\psi}_i^y(y) \;\middle|\; R^e\left(t, y, u^e(t,y), \dot{u}^e(t,y)\,, \ddot{u}^e(t,y)\right)_{\rho^y(y)}^{\mathcal{Y}} \right\rangle \\ \approx \\ \sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i^y(y_q) R^e\left(t, y_q, u^e(t,y_q), \dot{u}^e(t,y_q), \ddot{u}^e(t,y_q)\right)\end{gathered} \tag{7.33}$$

We repeatedly evaluate the deterministic element residuals for each value of quadrature node $y_q$ from the probabilistic domain $\mathcal{Y}$. As an input for evaluation of deterministic element residuals corresponding $y_q$, we need the deterministic state vectors and their time derivatives evaluated as

$$\begin{aligned} u^e(t, y_q) &\approx \sum_{i=1}^{N} U_i^e(t)\widehat{\psi}_i^y(y_q) \\ \dot{u}^e(t, y_q) &\approx \sum_{i=1}^{N} \dot{U}_i^e(t)\widehat{\psi}_i^y(y_q) \\ \ddot{u}^e(t, y_q) &\approx \sum_{i=1}^{N} \ddot{U}_i^e(t)\widehat{\psi}_i^y(y_q). \end{aligned} \tag{7.34}$$

In an iterative solution process such as Newton–Raphson method for solving nonlinear systems $U_i^e(t)$, $\dot{U}_i^e(t)$, and $\ddot{U}_i^e(t)$ are available as initial guesses. Thus all information required to form the stochastic residual by decomposing deterministic residuals on each basis term from stochastic space is readily available. The Algorithm 1 details the formation of stochastic residual by projecting deterministic residual on to the probabilistic basis elements.

### *7.3.2.2 Stochastic Element Jacobian*

Let the deterministic Jacobian of an element $e$ be $J^e\left(t, y, u^e(t, y), \dot{u}^e(t, y), \ddot{u}^e(t, y)\right)$ and the stochastic Jacobian be $\mathcal{J}^e\left(t, y, U^e(t), \dot{U}^e(t), \ddot{U}^e(t)\right)$. The process is similar to the residual assembly and the projection in stochastic domain is performed as

$$\begin{gathered} \left\langle \widehat{\psi}_i^y(y) \;\middle|\; J^e\left(t, y, u^e(t, y), \dot{u}^e(t, y)\,, \ddot{u}^e(t, y)\right) \;\middle|\; \widehat{\psi}_j^y(y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} \\ \approx \\ \sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i^y(y_q) J^e\left(t, y_q, u^e(t, y_q), \dot{u}^e(t, y_q), \ddot{u}^e(t, y_q)\right) \widehat{\psi}_j^y(y_q) \end{gathered} \tag{7.35}$$

The algorithm performing this is shown in Algorithm 2.

#### *7.3.2.3 Stochastic Initial Conditions*

The projection of initial conditions and their time derivatives is performed as follows

$$
\begin{aligned}
\left\langle \widehat{\psi}_i^y(y) \;\middle|\; u^e(0,y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} &\approx \sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i^y(y_q) u^e(0,y_q) \\
\left\langle \widehat{\psi}_i^y(y) \;\middle|\; \dot{u}^e(0,y) \right\rangle_{\rho^y(y)}^{\mathcal{Y}} &\approx \sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i^y(y_q) \dot{u}^e(0,\dot{y}_q)
\end{aligned}
\tag{7.36}
$$

This includes the case where the initial conditions also are dependent on random variables from probabilistic domain (uncertainty associated with initial conditions). The algorithm for this projection is shown in Algorithm 3. Usually for a second-order time dependent process, we require only upto first-time derivatives.

**Summary.** We presented the details of nonintrusive and semi-intrusive SGM for UQ, along with details of adjoint sensitivity analysis. We developed the stochastic FEM framework using simple extensions of the deterministic finite element framework. We discussed the formation of stochastic Jacobian matrices in detail with emphasis on optimizing quadrature evaluations alongside the sparsity and symmetry considerations. The application of the semi-intrusive approach for stochastic finite volume frameworks (FVM) is straight-forward and intuitive. In the case of stochastic finite volume method, we should work with cell-wise residuals and Jacobians, as opposed to element-wise residuals and Jacobians in the context of FEM. Recall that *cells* are the fundamental units of FVM computations, and *elements* are the fundamental units of FEM framework. In this thesis, the semi-intrusive method is demonstrated on FEM problems; Chatzimanolakis et al. [121] presents a similar exposition in the context of FVM for Reynolds-Averaged Navier–Stokes without adjoint sensitivity analysis.

**Algorithm 1** Algorithm for evaluating stochastic element residual from deterministic element residual.

```
Require: $pspace$                                                    # probabilistic space object
Require: $delem$                                                     # deterministic element object
function StochasticResidual(pspace, delem, $t, U^e, \dot{U}^e, \ddot{U}^e$)
    $N \leftarrow pspace.getNumBasisFunctions()$             # number of stochastic basis functions
    $N_{nodes} \leftarrow delem.getNumNodes()$                         # number of element nodes
    $N_{ddof} \leftarrow delem.getNumDof()$            # number of deterministic degrees of freedom
    $N_{sdof} \leftarrow N \times N_{ddof}$                  # number of stochastic degrees of freedom
    $\mathcal{R}^e = zeros(N_{sdof})$                        # set the stochastic residual to zero
    for $i = 1, N$ do                                            # loop over basis function set
        $pspace.InitializeQuadrature(i)$             # optimize number of quadrature points
        !  Perform projection using quadrature
        ${R_i}^e = zeros(N_{ddof})$                                  # zero vector for storage
        for $q = 1, Q$ do                                                   # quadrature loop
            $y_q, z_q, \alpha_q \leftarrow pspace.getQuadraturePointsWeight(q)$
            ${R_q}^e, {u_q}^e, \dot{u}_q^e, \ddot{u}_q^e = zeros(N_{ddof})$                       # zero vectors for storage
            !  Form state vectors as input for deterministic element
            for k = 1, N do                                      # loop over basis function set
                ${\psi_{q_k}}^z \leftarrow pspace.evaluateBasis(k, z_q)$          # evaluate k-th basis function
                ${u_q}^e \leftarrow {u_q}^e + U^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
                $\dot{u}_q^e \leftarrow \dot{u}_q^e + \dot{U}^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
                $\ddot{u}_q^e \leftarrow \ddot{u}_q^e + \ddot{U}^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
            end for
            ${\psi_{q_i}}^z \leftarrow pspace.evaluateBasis(i, z_q)$             # evaluate i-th basis function
            ${R_q}^e \leftarrow delem.getResidual(t, y_q, {u_q}^e, \dot{u}_q^e, \ddot{u}_q^e)$        # deterministic residual at $y_q$
            ${R_i}^e \leftarrow {R_i}^e + \alpha_q \times {\psi_{q_i}}^z \times {R_q}^e$                        # Equation (7.33)
        end for
        !  Order stochastic residuals nodewise
        for $ii = 1, N_{nodes}$ do
            $off \leftarrow ii \times N_{sdof}$
            $\mathcal{R}^e[off + i \times N_{ddof} : off + (i+1) \times N_{ddof}] \leftarrow {R_i}^e[ii \times N_{ddof} : (ii+1) \times N_{ddof}]$
        end for
    end for
    return $\mathcal{R}^e$
end function
```

**The philosophical awe - entropy and order:** We began this Chapter with Thomas Aquinas' quote on philosophy and awe. It appears that we use quadrature approximation of integrals in both sampling and projection methods. The question, out of curiosity, on how two fundamentally different approaches find application in the same context arises with awe. The thoughtful admiration lies in the realization that when we use sampling based approaches, the probabilistic domain $\mathcal{Y}$ is treated at the end, whereas when we perform projection we are indeed treating the probabilistic domain first and the temporal domain at the end! This simple and naive shuffled treatment of domains leads to remarkably different solution approaches to the problem of UQ. The mathematical trick of ordering has indeed manifested itself before – the adjoint and direct sensitivity analysis methods are yet another mentionable artifacts in the context of this thesis. This makes oneself humbled in the fact that the discovering new mathematical techniques, often, do not require new formulations, but rather depends on the perspective from which we approach the solution. It seems suitable that *informal trick of ordering* ought to be elevated to the revered status of being the *principle of ordering*, alongside the *principle of superposition* and the *principle separation of variables* in the context of solving differential equations in probabilistic-space-time.

**Algorithm 2** Algorithm for evaluating stochastic element Jacobian from deterministic element Jacobian.

```
Require: $pspace$                                              # probabilistic space object
Require: $delem$                                             # deterministic element object
function STOCHASTICJACOBIAN(pspace, delem, $t, U^e, \dot{U}^e, \ddot{U}^e$)
    $N \leftarrow pspace.getNumBasisFunctions()$        # number of stochastic basis functions
    $N_{nodes} \leftarrow delem.getNumNodes()$                         # number of element nodes
    $N_{ddof} \leftarrow delem.getNumDof()$          # number of deterministic degrees of freedom
    $N_{sdof} \leftarrow N \times N_{ddof}$                     # number of stochastic degrees of freedom
    $\mathcal{J}^e = zeros(N_{sdof}, N_{sdof})$                 # space for stochastic element Jacobian
    for $i = 1, N$ do                                  # loop over basis function set for rows
        for $j = 1, N$ do                           # loop over basis function set for columns
            $pspace.InitializeQuadrature(i, j)$                 # number of quadrature points
            !  Perform projection using quadrature
            ${J_{ij}}^e = zeros(N_{ddof}, N_{ddof})$                 # space for Jacobian block at $i, j$
            for $q = 1, Q$ do                                           # loop over quadrature
                $y_q, z_q, \alpha_q \leftarrow pspace.getQuadraturePointsWeight(q)$
                ${u_q}^e, \dot{u}^e_q, \ddot{u}^e_q = zeros(N_{ddof})$    # space for deterministic state variables at $y_q$
                !  Form state vectors as input for deterministic element
                for k = 1, N do
                    ${\psi_{q_k}}^z \leftarrow pspace.evaluateBasis(k, z_q)$       # evaluate k-th basis function
                    ${u_q}^e \leftarrow {u_q}^e + U^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
                    $\dot{u}^e_q \leftarrow \dot{u}^e_q + \dot{U}^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
                    $\ddot{u}^e_q \leftarrow \ddot{u}^e_q + \ddot{U}^e[k \times N_{ddof} : (k+1) \times N_{ddof}] \times {\psi_{q_k}}^z$
                end for
                ${\psi_{q_i}}^z \leftarrow pspace.evaluateBasis(i, z_q)$             # evaluate i-th basis function
                ${\psi_{q_j}}^z \leftarrow pspace.evaluateBasis(j, z_q)$             # evaluate j-th basis function
                ${J_q}^e = zeros(N_{ddof}, N_{ddof})$            # space for deterministic Jacobian at $y_q$
                ${J_q}^e \leftarrow delem.getJacobian(t, y_q, {u_q}^e, \dot{u}^e_q, \ddot{u}^e_q)$  # deterministic Jacobian at $y_q$
                ${J_{ij}}^e \leftarrow {J_{ij}}^e + \alpha_q \times {\psi_{q_i}}^z \times {J_q}^e \times {\psi_{q_j}}^z$                          # Equation (7.35)
            end for
            !  Order stochastic jacobians nodewise
            for $ii = 1, N_{nodes}$ do                                     # outer loop over nodes
                for $jj = 1, N_{nodes}$ do                                 # inner loop over nodes
                    $istart \leftarrow ii \times N_{sdof} + i \times N_{ddof}$                        # row start index
                    $iend \leftarrow jj \times N_{sdof} + (i+1) \times N_{ddof}$                       # row end index
                    $jstart \leftarrow jj \times N_{sdof} + j \times N_{ddof}$                   # column start index
                    $jend \leftarrow jj \times N_{sdof} + (j+1) \times N_{ddof}$                   # column end index
                    $\mathcal{J}^e[istart : iend, jstart : jend] \leftarrow {J_{ij}}^e[ii \times N_{ddof} : (ii+1) \times N_{ddof}, jj \times N_{ddof} : (jj+1) \times N_{ddof}]$
                end for
            end for
        end for
    end for
    return $\mathcal{J}^e$
end function
```

**Algorithm 3** Algorithm for evaluating stochastic element initial conditions from deterministic initial conditions.

```
Require: $pspace$                                                  # probabilistic space object
Require: $delem$                                                   # deterministic element object
function StochasticInitialCondition(pspace, delem, $t, U^e, \dot{U}^e$)
    $N \leftarrow pspace.getNumBasisFunctions()$                    # number of stochastic basis functions
    $N_{nodes} \leftarrow delem.getNumNodes()$                      # number of element nodes
    $N_{ddof} \leftarrow delem.getNumDof()$                         # number of deterministic degrees of freedom
    $N_{sdof} \leftarrow N \times N_{ddof}$                         # number of stochastic degrees of freedom
    $U^e, \dot{U}^e = zeros(N_{sdof})$                              # zero vectors for storage
    for $i = 1, N$ do                                               # loop over basis function set
        $pspace.InitializeQuadrature(i)$                            # number of quadrature points
        ! Perform projection using quadrature
        for $q = 1, Q$ do                                           # loop over quadrature points
            $y_q, z_q, \alpha_q \leftarrow pspace.getQuadraturePointsWeight(q)$
            ${\psi_{q_i}}^z \leftarrow pspace.evaluateBasis(i, z_q)$    # evaluate i-th basis function
            ${u_q}^e, \dot{u}_q^e \leftarrow delem.getInitialConditions(t, y_q)$    # deterministic ICs at $y_q$
            ${U_i}^e \leftarrow {U_i}^e + \alpha_q \times {\psi_{q_i}}^z \times {u_q}^e$    # Equation (7.36)
            $\dot{U}_i^e \leftarrow \dot{U}_i^e + \alpha_q \times {\psi_{q_i}}^z \times \dot{u}_q^e$    # Equation (7.36)
        end for
        ! Order stochastic initial conditions nodewise
        for $ii = 1, N_{nodes}$ do                                  # loop overs nodes in the element from mesh
            $off \leftarrow ii \times N_{sdof}$                     # find the start index
            $U^e[off + i \times N_{ddof} : off + (i+1) \times N_{ddof}] \leftarrow {U_i}^e[ii \times N_{ddof} : (ii+1) \times N_{ddof}]$
            $\dot{U}^e[off + i \times N_{ddof} : off + (i+1) \times N_{ddof}] \leftarrow \dot{U}_i^e[ii \times N_{ddof} : (ii+1) \times N_{ddof}]$
        end for
    end for
    return $U^e$, $\dot{U}^e$
end function
```

# CHAPTER 8
# UNCERTAINTY PROPAGATION AND SENSITIVITY ANALYSIS OF STATIC AND TIME DEPENDENT SYSTEMS

**Introduction.** The goal of this chapter is to demonstrate the computation of probabilistic moments and their derivatives using the semi-intrusive SGM, and to compare the moments computed to that of sampling-based collocation method. We begin with stochastic analysis of a static system, and proceed to the analysis of time dependent first and second-order systems, and finally to flexible multibody dynamics systems.

## 8.1 Static Spring

We consider the static deflection of a spring subject to a force $f$ as the physical problem of interest. The metric of interest is the potential energy of the system, and the design variable is the stiffness constant $k$, that is $\xi := k$.

### 8.1.1 Deterministic System

The system is mathematically modeled as an algebraic equation:

Physical Analysis

$\underset{u(\xi)}{\text{solve}}$ $\quad R(\xi, u(\xi)) := k \cdot u(\xi) - f = 0$ $\quad$ (residual)

evaluate $\quad F(\xi, u(\xi)) := \frac{1}{2} k u(\xi)^2$ $\quad$ (metric of interest)

Sensitivity Analysis

evaluate $\quad \dfrac{dF(\xi, u(\xi))}{d\xi}$ $\quad$ (derivative of function)

The solution of state variables $u(\xi)$ is obtained solving a linear system as

$$u(\xi) = \frac{f}{k}. \tag{8.1}$$

The metric of interest (potential energy) is

$$F(\xi, u(\xi)) = \frac{1}{2} \times k \times \left(\frac{f}{k}\right)^2 = \frac{1}{2}\frac{f^2}{k} \tag{8.2}$$

The derivative of function of interest (energy) with respect to stiffness constant $k$ is

$$\frac{dF(\xi, u(\xi))}{d\xi} = -\frac{1}{2}\frac{f^2}{k^2} \tag{8.3}$$

The metric (8.2) and its design variable derivative (8.3) can be used to solve optimization problems using gradient-based algorithms, to identify the critical $\xi^\star$ that minimizes or maximizes the potential energy. For the purposes of demonstration, let the right hand side $f = \pi$ and stiffness constant $k = \frac{\pi}{2}$ which evaluates to

$$F = \pi \quad \text{and} \quad \frac{dF}{d\xi} = -2. \tag{8.4}$$

### 8.1.2 Stochastic System with Uncertain System Parameter

Now let us explore the changes in the metric and its derivative in the presence of uncertainties. Let us assume that the stiffness parameter is uncertain and dependent on random variable such that $k := k(y)$, where $y$ is a random variable from stochastic domain $\mathcal{Y}$. The random variable adds a stochastic dimension to the *deterministic linear algebraic equation*, which results in a *stochastic linear algebraic equation.* Mathematically, the setup of the stochastic problem we intend to solve is given as follows.

Physical Analysis

$$\begin{array}{lll} \underset{u(y)}{\text{solve}} & R(y,u(y)) := k(y)\cdot u(y) - f = 0 & \text{(stochastic residual)} \\ \text{evaluate} & \mathbb{E}[F(y,u(y))] := \mathbb{E}\left[\frac{1}{2}k(y)u(y)^2\right] & \text{(mean)} \\ & \mathbb{V}[F(y,u(y))] := \mathbb{V}\left[\frac{1}{2}k(y)u(y)^2\right] & \text{(variance)} \\ & \mathbb{S}[F(y,u(y))] := \ \mathbb{S}\left[\frac{1}{2}k(y)u(y)^2\right] & \text{(standard deviation)} \end{array}$$

Sensitivity Analysis

$$\begin{array}{ll} \dfrac{d\mathbb{E}[F(y(\xi),u(y(\xi)))]}{d\xi} & \text{(derivative of mean)} \\ \dfrac{d\mathbb{V}[F(y(\xi),u(y(\xi)))]}{d\xi} & \text{(derivative of variance)} \\ \dfrac{d\mathbb{S}[F(y(\xi),u(y(\xi)))]}{d\xi} & \text{(derivative of std. dev.)} \end{array}$$

The computation of these quantities allows us to solve an optimization under uncertainty problem, to identify a critical $\xi^\star$ that provides robustness and reliability. In this stochastic case the design variable $\xi$ is chosen to be the mean stiffness $\mu_k$. Let us discuss the computation of the probabilistic moments of the metric and its derivative using sampling and projection.

**(1) Sampling Method:** The first moment and its derivative are computed using sampling as:

$$\mathbb{E}\left[F(y,u(y))\right] \approx \sum_{q=1}^{Q} \alpha_q^y \left(\frac{1}{2}\frac{f^2}{k_q}\right) \quad \text{and} \quad \mathbb{E}\left[\frac{dF(y,u(y))}{d\xi}\right] \approx \sum_{q=1}^{Q} \alpha_q^y \left(\frac{1}{2}\frac{f^2}{{k_q}^2}\right). \tag{8.5}$$

where $k_q$ is the $q-$th quadrature node whose corresponding weight is $\alpha_q^y$. The variance $\mathbb{V}\left[F(y,u(y))\right]$ and standard deviation $\mathbb{S}\left[F(y,u(y))\right]$ are computed following the steps outlined in Section 7.1.

**(2) Projection Method:** When we apply the projection method, we expect an expanded linear system, featuring bigger matrices and vectors. Using projection method, the $i-$th right hand side is formed as

$$f_i \approx \sum_{q=1}^{Q} \alpha_q^y \times \widehat{\psi}_i(y_q) \times f \tag{8.6}$$

and the $(i,j)-th$ Jacobian matrix entries are formed as

$$\begin{aligned} \mathcal{J}_{i,j} &\approx \sum_{q=1}^{Q} \alpha_q^y \times \widehat{\psi}_i(y_q) \times J(y_q, u(y_q)) \times \widehat{\psi_j}(y_q) \\ &= \sum_{q=1}^{Q} \alpha_q^y \times \widehat{\psi}_i(y_q) \times k_q \times \widehat{\psi_j}(y_q) \end{aligned} \tag{8.7}$$

This results in the linear system

$$\begin{bmatrix} \mathcal{J}_{1,1} & \mathcal{J}_{1,2} & \cdots & \mathcal{J}_{1,N} \\ \mathcal{J}_{2,1} & \mathcal{J}_{2,2} & \cdots & \mathcal{J}_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{J}_{N,1} & \mathcal{J}_{N,2} & \cdots & \mathcal{J}_{N,N} \end{bmatrix} \left\{ \begin{matrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{matrix} \right\} = \left\{ \begin{matrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{matrix} \right\} \tag{8.8}$$

In this case the Jacobian $\mathcal{J}$ is a tridiagonal matrix since the polynomial degree of the stiffness parameter is one, giving rise to one band on either side of the diagonal. The decomposition coefficients $u_i$ are determined by solving the linear system (8.8). The probabilistic moments of $F$ and its derivatives are obtained using the steps outlined in Section 7.2.

**Complex-step verification of stochastic adjoint derivatives:** First, we ensure the consistency of the derivative of moments of $F$ with respect to $\xi = \mu_k$ using the complex step method. This test is conducted with normal, exponential and uniform probability distributions of the random variable $y$. Table 8.1 lists the adjoint derivatives computed using projection with that of the complex step method. An exact match of values can be noticed, which verifies the validity of the adjoint derivatives computed using the semi-intrusive projection.

*Table 8.1: The complex-step verification of stochastic adjoint derivatives with the number of terms in orthonormal basis $N = 7$.*

| Distribution | Type | $\frac{d\mathbb{E}[F]}{d\xi}$ | $\frac{d\mathbb{V}[F]}{d\xi}$ | $\frac{d\mathbb{S}[F]}{d\xi}$ |
|---|---|---|---|---|
| Exponential | adjoint | -1.68733320815981 | -0.11533024885544 | -0.25255516821879 |
| $\mathcal{E}(\frac{\pi}{2}, 0.1 \cdot \frac{\pi}{2})$ | complex | -1.68733320815981 | -0.11533024885544 | -0.25255516821879 |
| Normal | adjoint | -2.06323129529018 | -0.28546603338653 | -0.43563400146749 |
| $\mathcal{N}(\frac{\pi}{2}, 0.1 \cdot \frac{\pi}{2})$ | complex | -2.06323129529018 | -0.28546603338653 | -0.43563400146749 |
| Uniform | adjoint | -2.02020202020202 | -0.08564848260089 | -0.23436946202514 |
| $\mathcal{U}(0.9 \cdot \frac{\pi}{2}, 1.1 \cdot \frac{\pi}{2})$ | complex | -2.02020202020202 | -0.08564848260089 | -0.23436946202514 |

**Comparison of sampling- and projection-based metrics:** Now we compare the metrics and derivatives obtained using the semi-intrusive projection with that of the quadrature sampling. The use of more number of quadrature samples (for collocation) and more number of basis terms (for projection) will, in general, produce a better estimate of probabilistic moments and derivatives. However, these are not known apriori at the start of stochastic analysis process. In this section, a notional comparison is made with the number of samples $Q = 15$ and the number of terms in orthonormal basis $N = 7$, and the values are listed in Table 8.2. We highlight the mismatching digits between the two methods in boldface font. Recall that for the sampling method we solve the deterministic linear system of size $1 \times 1$, $Q$ times, whereas in the case of the projection a linear system of size $7 \times 7$ is solved. Similarly for the adjoint derivatives, a smaller adjoint system is solved $Q$ times with sampling, however a larger adjoint system is solved once with projection.

*Table 8.2: Probabilistic moments of the energy function and its design variable derivative with the number of samples $Q = 15$ and the number of terms in orthonormal basis $N = 7$.*

| | Function | Value | Derivative | Value |
|---|---|---|---|---|
| deterministic | $F$ | $\pi$ | $\frac{dF}{d\xi}$ | -2 |
| sampling | $\mathbb{E}[F]$ | 2.87654697**243427** | $d\mathbb{E}[F]/d\xi$ | -1.6873332**1204238** |
| projection | $\mathbb{E}[F]$ | 2.87654697**182409** | $d\mathbb{E}[F]/d\xi$ | -1.6873332**0815981** |
| sampling | $\mathbb{V}[F]$ | 0.0521331**6321809** | $\mathbb{V}[F]/d\xi$ | -0.115330**35551488** |
| projection | $\mathbb{V}[F]$ | 0.0521331**4756883** | $d\mathbb{V}[F]/d\xi$ | -0.115330**24885544** |
| sampling | $\mathbb{S}[F]$ | 0.2283268**7800188** | $d\mathbb{S}[F]/d\xi$ | -0.252555**36388040** |
| projection | $\mathbb{S}[F]$ | 0.2283268**4373246** | $d\mathbb{S}[F]/d\xi$ | -0.252555**16821879** |

## 8.2 Linear Time Dependent Systems

In this section, we demonstrate the uncertainty propagation using sampling and projection on the following linear time dependent systems:

- one degree of freedom decay model (first-order ODE)
- one degree of freedom spring mass damper system (second-order ODE)
- three degrees of freedom spring mass series (second-order ODE)
- two degrees of freedom pitching and plunging airfoil system (second-order ODE)

### 8.2.1 First-Order Decay Model

We consider a simple first-order ordinary differential equation (ODE). The statistical moments (mean and variance) of the solution to the ODE are found using: (a) analytical formulae, (b) quadrature sampling and (c) projection techniques. We study this problem with three distributions types (normal, uniform and exponential) for the decay parameter.

#### *8.2.1.1 Deterministic and Stochastic Problem Setup*

Consider the following first-order differential equation with prescribed initial condition $u_0$ and constant decay parameter $\lambda$

$$\begin{aligned} \frac{du(t)}{dt} + \lambda\ u(t) = 0 \quad &\in \ \mathcal{T} \\ u(0) = u_0 \quad &\in \ \partial\mathcal{T} \end{aligned} \tag{8.9}$$

The analytical solution to the differential equation is $u(t) = u_0\,\mathrm{e}^{-\lambda t}$. Let us assume that the decay parameter is uncertain $\lambda := \lambda(y)$, where $y$ is a random variable from stochastic domain $\mathcal{Y}$. This adds a stochastic dimension to the differential equation resulting in the stochastic differential equation

$$\begin{aligned} \frac{du(t,y)}{dt} + \lambda(y)\ u(t,y) = 0 \quad &\in \ \mathcal{T}\otimes\mathcal{Y} \\ u(0,y) = u_0 \quad &\in \ \partial\mathcal{T}\otimes\mathcal{Y} \end{aligned} \tag{8.10}$$

Notice the product nature of temporal and stochastic domains forming the stochastic-temporal domain. Rather than individual solutions $u(t,y) = u_0\,\mathrm{e}^{-\lambda(y)t}$ for each realization of the random variable, we are interested in statistical moments of the solution, namely the mean $\mathbb{E}[u(t,y)]$ (first moment) and variance $\mathbb{V}[u(t,y)]$ (second moment).

#### *8.2.1.2 Analytical Moments for Gaussian Distribution*

Let $y$ be normally distributed with mean $\mu$ and standard deviation $\sigma$ as $y \sim \mathcal{N}(\mu,\sigma)$. The mean (first moment) is

$$\begin{aligned} \mathbb{E}[u(t,y)] &= \int_{-\infty}^{\infty} u(t,y)\,\underbrace{\frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}}_{\rho_n^y(y)}\,dy = \int_{-\infty}^{\infty} u_0\mathrm{e}^{-yt}\frac{\mathrm{e}^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}}\,dy \\ &= \exp\left(\frac{1}{2}\sigma^2t^2 - \mu t\right) \end{aligned} \tag{8.11}$$

The variance (second moment) is

$$\begin{aligned}\mathbb{V}[u(t,y)] &= \mathbb{E}[u^2(t,y)] - \mathbb{E}[u(t,y)]^2 \\ &= \exp\left(2\sigma^2 t^2 - 2\mu t\right) - \left(\exp\left(\frac{1}{2}\sigma^2 t^2 - \mu t\right)\right)^2\end{aligned} \tag{8.12}$$

*8.2.1.3 Stochastic Quadrature Sampling for Normal Distribution*

Stochastic sampling technique uses repeated solutions of the differential equation (8.10) to perform integration in the stochastic domain. The decay parameter value $\lambda_q = \lambda(y_q)$ depends on the random variable $y_q$ and its corresponding distribution type. The stochastic ODE

$$\begin{aligned}\frac{du(t,y_q)}{dt} + \lambda_q u(t,y_q) = 0 \quad &\in \quad \mathcal{T} \otimes \mathcal{Y} \\ u(0,y_q) = u_0 \quad &\in \quad \partial\mathcal{T} \otimes \mathcal{Y}\end{aligned} \tag{8.13}$$

is solved for each $\lambda_q = \lambda(y_q)$ and the solutions $u(t,y_q)$ are stored. The moments of the solution are computed as

$$\begin{aligned}\mathbb{E}[u(t,y)] &= \sum_{q=1}^{Q} \alpha_q^y u(t,y_q) \\ \mathbb{V}[u(t,y)] &= \sum_{q=1}^{Q} \alpha_q^y u^2(t,y_q) - \mathbb{E}[u(t,y)]^2\end{aligned} \tag{8.14}$$

Figure 8.1 compares mean and variance computed using the stochastic sampling method with the analytical mean and variance for increasing number of samples from the stochastic domain. It can be seen that the first moment converges faster than the second moment to the analytical value.

*8.2.1.4 Stochastic Projection for Normal Distribution*

Let us explore the solution of the stochastic problem (8.10) applying the principle of projection in stochastic space, with normally distributed decay parameter.

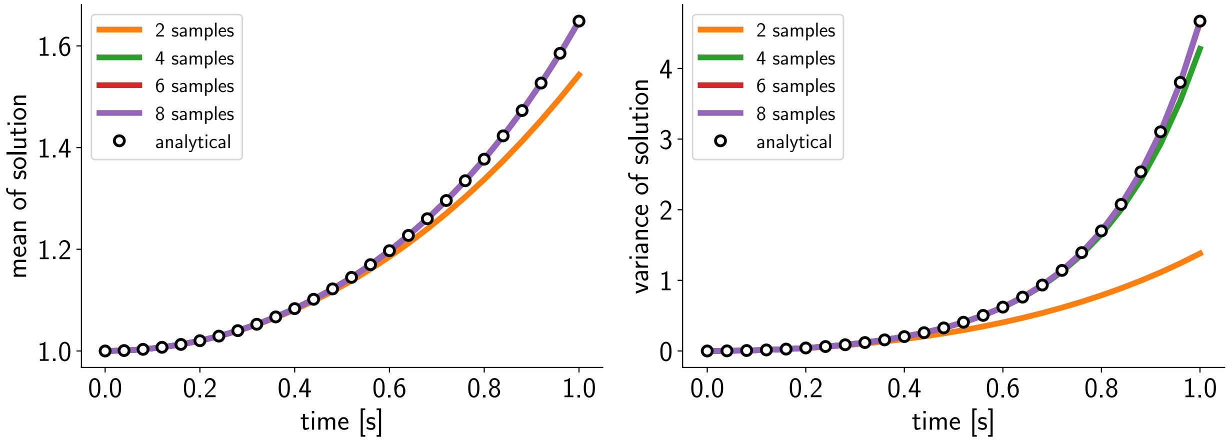


*Figure 8.1: Comparison of the sampling-based mean and variance of the solution with analytical moments for $\mu_\lambda = 0$ and $\sigma_\lambda = 1$.*

**The representation and decomposition of decay parameter:** The decay parameter is a function of random variable $y$ and assumes a spectral expansion:

$$\lambda(y) = \sum_{j=1}^{N} \lambda_j \widehat{\psi}_j^y(y) \in \mathcal{Y} \tag{8.15}$$

where the decomposition coefficients are found using projection as follows

$$\lambda_j = \left\langle \lambda(y) \,\middle|\, \widehat{\psi}_j^y(y) \right\rangle_{\rho_n^y(y)}^{\mathcal{Y}}, \quad \forall j = 1, \ldots, N. \tag{8.16}$$

and $\widehat{\psi}_j^y(y)$ refers to the Hermite polynomials $\widehat{H}_{j-1}^y(y)$. In vector notation, we get:

$$\Lambda = \begin{Bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \\ \lambda_N \end{Bmatrix} = \begin{Bmatrix} \langle \lambda(y) \mid \widehat{\psi}_1^y(y) \rangle_{\rho_n^y}^{\mathcal{Y}} \\ \langle \lambda(y) \mid \widehat{\psi}_2^y(y) \rangle_{\rho_n^y}^{\mathcal{Y}} \\ \langle \lambda(y) \mid \widehat{\psi}_3^y(y) \rangle_{\rho_n^y}^{\mathcal{Y}} \\ \vdots \\ \langle \lambda(y) \mid \widehat{\psi}_N^y(y) \rangle_{\rho_n^y}^{\mathcal{Y}} \end{Bmatrix} = \begin{Bmatrix} \mu_\lambda \\ \sigma_\lambda \\ 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{8.17}$$

**(a) The representation and decomposition of initial conditions:** The functional form of the initial conditions in the stochastic ODE (8.10) is

$$u(0,y) = u_0 \quad \in \quad \partial\mathcal{T} \otimes \mathcal{Y} \tag{8.18}$$

In this case, we have considered a constant $u_0$ as the initial state, but it can be a general function of random variable. Let the right hand side of the initial conditions be provided as abstract functions of random variable $y$

$$u(0,y) = g(y) \quad \in \quad \partial\mathcal{T} \otimes \mathcal{Y} \tag{8.19}$$

Similar to the stochastic parameter $\lambda(y)$, the initial condition assumes a spectral representation as

$$g(y) = \sum_{j=1}^{N} g_j \widehat{\psi}^y_j(y) \in \mathcal{Y} \tag{8.20}$$

where $g_j$ are the coefficients of decomposition of initial condition in corresponding stochastic vector space $\mathcal{Y}$ obtained using inner products as

$$g_j = \left\langle g(y) \;\middle|\; \widehat{\psi}^y_j(y) \right\rangle^{\mathcal{Y}}_{\rho^y_n(y)} \tag{8.21}$$

In vector notation,

$$g = \begin{Bmatrix} g_1 \\ g_2 \\ \vdots \\ g_N \end{Bmatrix} = \begin{Bmatrix} \langle g(y) \mid \widehat{\psi}_1 y(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \\ \langle g(y) \mid \widehat{\psi}^y_2(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \\ \vdots \\ \langle g(y) \mid \widehat{\psi}^y_N(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \end{Bmatrix} = \begin{Bmatrix} \langle u_0 \mid \widehat{\psi}^y_1(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \\ \langle u_0 \mid \widehat{\psi}^y_2(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \\ \vdots \\ \langle u_0 \mid \widehat{\psi}^y_N(y)\rangle^{\mathcal{Y}}_{\rho^y_n} \end{Bmatrix} = \begin{Bmatrix} u_0 \\ 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{8.22}$$

**(b) The representation and decomposition of the state variables:** The stochastic-temporal field $u(t,y)$ is represented as the following finite spectral summation

$$u(t,y) = \sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y) \quad \text{and} \quad \dot{u}(t,y) = \sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y) \tag{8.23}$$

where $u_i(t)$ and $\dot{u}_i(t)$ are the decomposition coefficients that are time dependent functions.

**(c) The stochastic ODE:** By applying (8.23) to (8.10) we obtain the stochastic ODE as

$$\mathcal{R} := \left(\sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y)\right) + \lambda(y)\left(\sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y)\right) = 0 \tag{8.24}$$

The stochastic ODE takes the following matrix form written out explicitly

$$\begin{bmatrix} \langle\widehat{\psi}_1^y(y) \mid \widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_1^y(y) \mid \widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_1^y(y) \mid \widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \\ \langle\widehat{\psi}_2^y(y) \mid \widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_2^y(y) \mid \widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_2^y(y) \mid \widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \\ \vdots & \vdots & \ddots & \vdots \\ \langle\widehat{\psi}_N^y(y) \mid \widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_N^y(y) \mid \widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_N^y(y) \mid \widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \end{bmatrix} \begin{Bmatrix} \dot{u}_1(t) \\ \dot{u}_2(t) \\ \vdots \\ \dot{u}_N(t) \end{Bmatrix}$$

$$+$$

$$\begin{bmatrix} \langle\widehat{\psi}_1^y(y)|\lambda(y)|\widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_1^y(y)|\lambda(y)|\widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_1^y(y)|\lambda(y)|\widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \\ \langle\widehat{\psi}_2^y(y)|\lambda(y)|\widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_2^y(y)|\lambda(y)|\widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_2^y(y)|\lambda(y)|\widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \\ \vdots & \vdots & \ddots & \vdots \\ \langle\widehat{\psi}_N^y(y)|\lambda(y)|\widehat{\psi}_1^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \langle\widehat{\psi}_N^y(y)|\lambda(y)|\widehat{\psi}_2^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} & \dots & \langle\widehat{\psi}_N^y(y)|\lambda(y)|\widehat{\psi}_N^y(y)\rangle_{\rho_n^y}^{\mathcal{Y}} \end{bmatrix} \begin{Bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_N(t) \end{Bmatrix} \tag{8.25}$$

$$=$$

$$\begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{Bmatrix}$$

The analytical evaluation of these inner products leads to the stochastic ODE derived in explicit form:

$$\begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix} \begin{Bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \vdots \\ \dot{u}_{N-1} \\ \dot{u}_N \end{Bmatrix} + \begin{bmatrix} \mu & \sigma\sqrt{1} & \dots & 0 & 0 \\ \sigma\sqrt{1} & \mu & \sigma\sqrt{2} & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \sigma\sqrt{N-1} & \mu & \sigma\sqrt{N} \\ 0 & 0 & \dots & \sigma\sqrt{N} & \mu \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{Bmatrix} \tag{8.26}$$

**Remark on Sparsity.** When $\lambda$ is a constant function of $y$, then the Jacobian is diagonal. If $\lambda$ is a linear function of $y$, then the the Jacobian is tridiagonal (this case). If $\lambda$ is a quadratic function of $y$, then the Jacobian is pentadiagonal.

**Remark on Explicit Stochastic Equations.** For the simple stochastic ODE (8.10), we are able to derive the explicit form using the aforementioned steps. This is the approach taken for uncertainty propagation via the stochastic Galerkin method. However, when the governing equations take complicated expressions and contain nonlinear terms, the application of these steps become algebraically cumbersome. Krenk and Gutirez [59] point out that the projection-based SGM for problems involving nonlinearities have not yet reached a mature stage. The semi-intrusive stochastic Galerkin method presented in Section 7.2 can be used to address this difficulty.

**(d) The evaluation of Statistical moments:** The mean of the state variables is

$$\mathbb{E}[u(t,y)] = u_1(t) \tag{8.27}$$

and the variance is

$$\mathbb{V}[u(t,y)] = \sum_{i=2}^{N} u_i^2(t) \tag{8.28}$$

We use projection method described above for parameter values $\mu_\lambda = 0$ and $\sigma_\lambda = 1$. Figure 8.2 shows the comparison of mean and variance computed using the projection method with the analytical moments for increasing number of terms in the spectral expansion. Figure 8.3 shows the rate of convergence of mean and variance to analytical solutions.

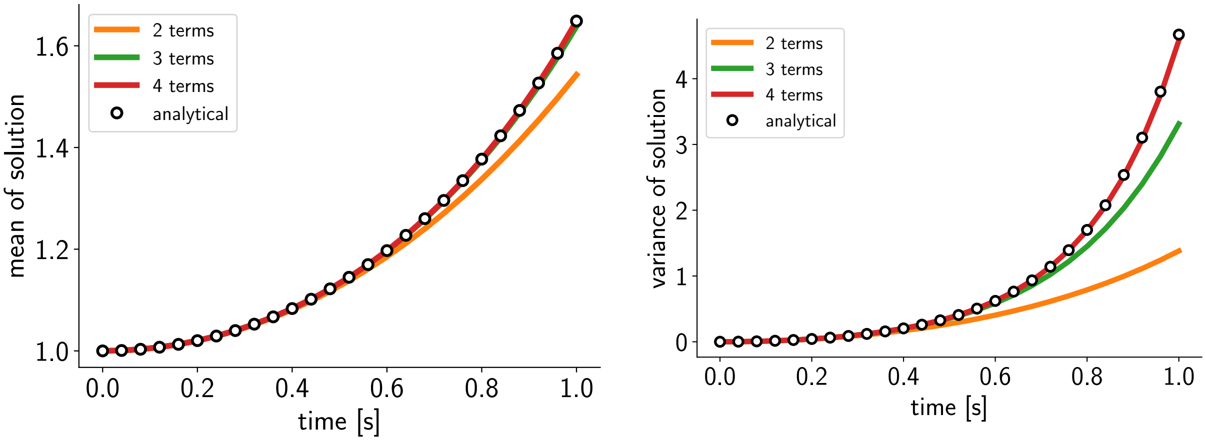


*Figure 8.2: Comparison of mean and variance of solution for selected number of terms in polynomial expansion.*

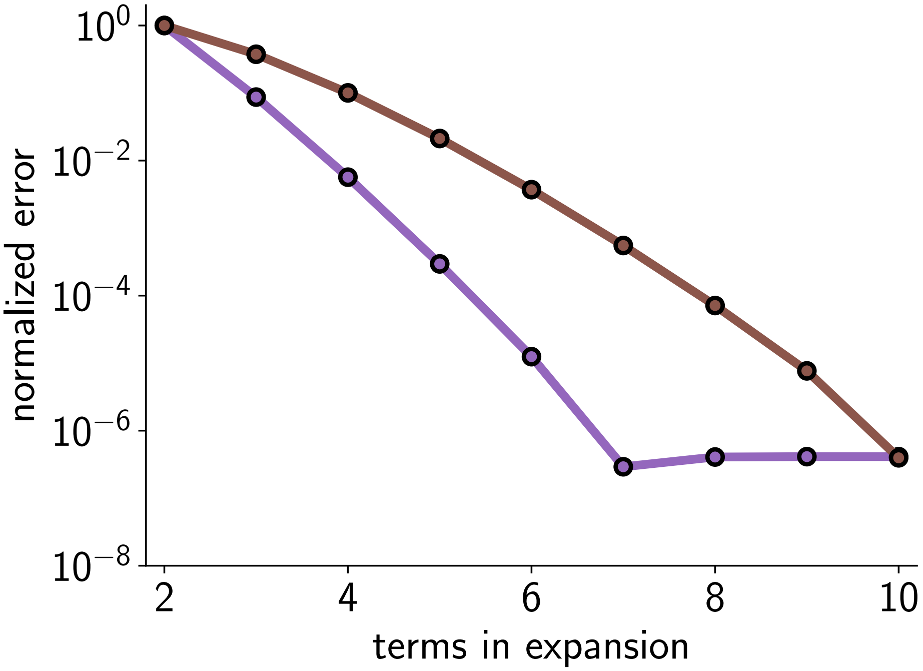


*Figure 8.3: The convergence of the mean and variance to the analytical values.*

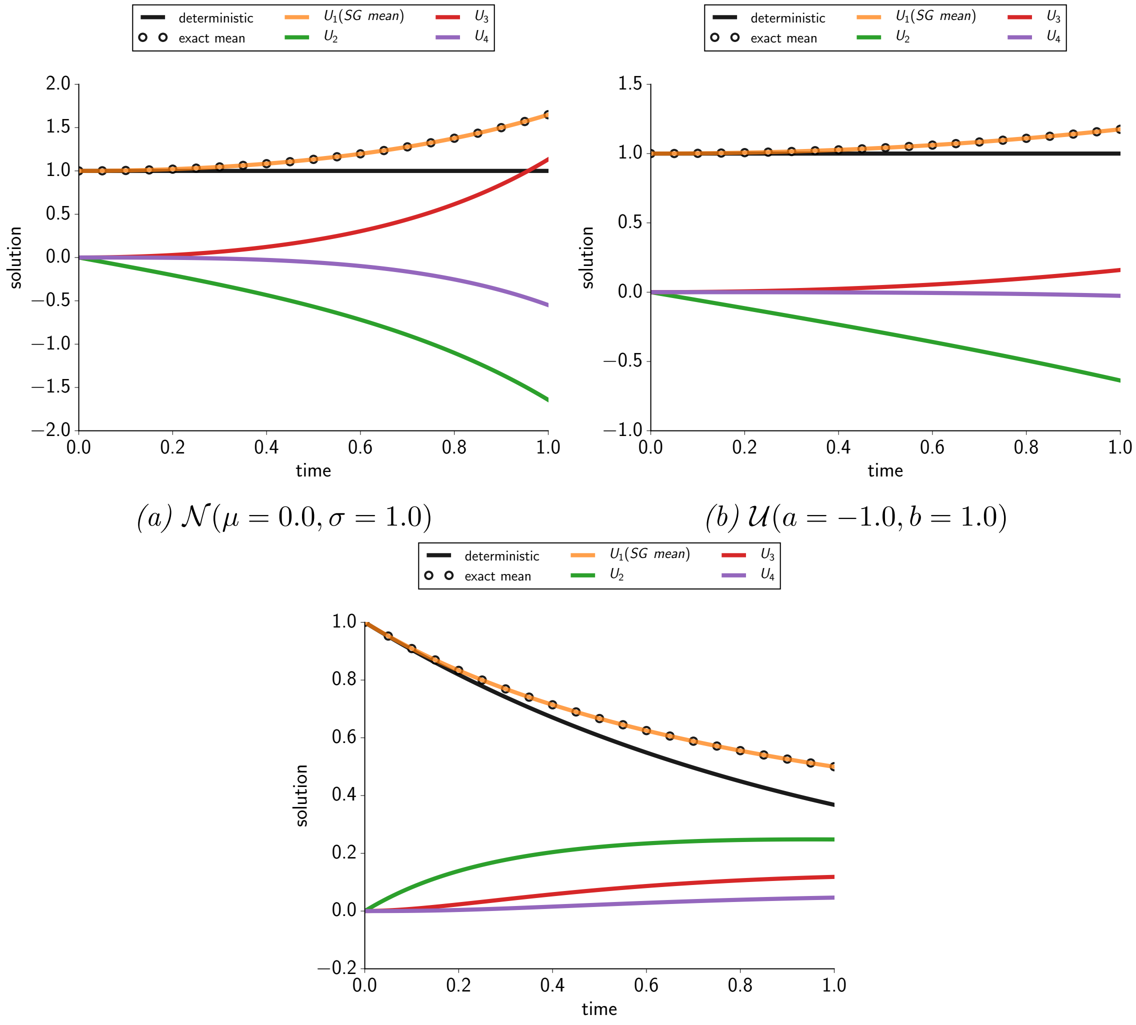
#### 8.2.1.5 Verification of Probabilistic Modes (Normal, Uniform and Exponential)

Here, we solve the stochastic problem (8.10) with three assumed distributions:

1. Normal $\mathcal{N}(\mu = 0, \sigma = 1)$,

2. Uniform $\mathcal{U}(a = -1, b = 1)$ and

3. Exponential $\mathcal{E}(\mu = 0, \beta = 1)$.



(a) $\mathcal{N}(\mu = 0.0, \sigma = 1.0)$

(b) $\mathcal{U}(a = -1.0, b = 1.0)$

(c) $\mathcal{E}(\mu = 0.0, \beta = 1.0)$

*Figure 8.4: The probabilistic modes computed using the stochastic Galerkin method along with the analytical mean and deterministic solutions for different distribution types.*

The stochastic Galerkin solution is obtained using a stochastic space spanned by $N = 4$ terms in the orthonormal basis set. Figure 8.4 plots the solution of the stochastic ODE and the four probabilistic modes computed, along with deterministic and analytical mean solutions from [72]. It can be seen that the mean of the solution is different than the deterministic solution in the absence of uncertainties, which shows that the system behavior can be far different than expected in the presence of uncertainties. The mean solution computed using SGM is matching with the analytical solution available to this simple benchmark problem reported in [72].

### 8.2.2 Natural Vibration of Spring Mass Damper System

We consider the following second-order differential equation with prescribed initial conditions $u_0$, $\dot{u}_0$ and constant system parameters $m$, $c$ and $k$

$$\begin{aligned} m\frac{d^2u(t)}{dt^2} + c\frac{du(t)}{dt} + ku(t) = 0 \quad &\in \quad \mathcal{T} \\ u(0) = u_0 \quad &\in \quad \partial\mathcal{T} \\ \dot{u}(0) = \dot{u}_0 \quad &\in \quad \partial\mathcal{T} \end{aligned} \tag{8.29}$$

Let us assume that the mass $m := m(y_1)$, damping constant $c := c(y_2)$ and stiffness constant $k := k(y_3)$ where $y_1, y_2$ and $y_3$ are independent random variables from three-dimensional stochastic domain $\mathcal{Y}^3$. This dependence of the system coefficients $m$, $c$ and $k$ on probabilistic random variables results in the stochastic differential equation

$$\begin{aligned} m(y_1)\frac{d^2u(t,y)}{dt^2} + c(y_2)\frac{du(t,y)}{dt} + k(y_3)u(t,y) = 0 \quad &\in \quad \mathcal{T} \otimes \mathcal{Y}^3 \\ u(0,y) = u_0 \quad &\in \quad \partial\mathcal{T} \otimes \mathcal{Y}^3 \\ \dot{u}(0,y) = \dot{u}_0 \quad &\in \quad \partial\mathcal{T} \otimes \mathcal{Y}^3 \end{aligned} \tag{8.30}$$

where $y = [y_1, y_2, y_3] \in \mathcal{Y}^3$ is a vector-valued random variable from stochastic space. Let the random variables be modeled as follows

$$\begin{aligned} y_1 &\sim \mathcal{E}(\mu = 4.00, \quad \beta = 1.00) \\ y_2 &\sim \mathcal{U}(a = 0.25, \quad b = 0.75) \\ y_3 &\sim \mathcal{N}(\mu = 5.00, \quad \sigma = 0.50) \end{aligned} \tag{8.31}$$

and the initial conditions be $u_0 = -0.5$ and $\dot{u}_0 = 1$.

#### *8.2.2.1 Details of Stochastic Galerkin Projection*

The formation of multivariate basis functions from univariate basis functions as well as the multivariate quadrature rules are essential for uncertainty propagation.

**Trivariate Quadrature Rule** Let $Q_1$, $Q_2$ and $Q_3$ be the number of quadrature points chosen for the corresponding one-dimensional Gauss–Laguerre, Gauss–Legendre and Gauss–Hermite quadrature, based on the resultant degree of the integrand and applying the exactness theorem of polynomial integration using quadrature. We can construct a three dimensional quadrature rule using tensor multiplication of one-dimensional quadrature rule for numerical approximation of inner products. Consider two functions $f(y)$ and $g(y)$ where $y \in \mathcal{Y}^3$

$$\begin{aligned} \Big\langle f(y) \;\Big|\; g(y) \Big\rangle_{\rho^y}^{\mathcal{Y}^3} &\approx \sum_{i=1}^{Q_1} \sum_{j=1}^{Q_2} \sum_{k=1}^{Q_3} f(y_{i,1}, y_{j,2}, y_{k,3}) \alpha_i^y \alpha_j^y \alpha_k^y g(y_{i,1}, y_{j,2}, y_{k,3}) \\ &= \sum_{l=1}^{Q} f(y_l) \alpha_l^y g(y_l) \end{aligned} \tag{8.32}$$

where the total number of quadrature points $Q = Q_1 \times Q_2 \times Q_3$, the weights $\alpha_l^y = \alpha_i^y \alpha_j^y \alpha_k^y$ and trivariate ordered nodes $y_l = [y_{i,1}, y_{j,2}, y_{k,3}]$. Note that the new weights must add upto unity as it was the case with univariate quadrature

$$\sum_{l=1}^{Q} \alpha_l^y = 1. \tag{8.33}$$

This condition can be used as a simple test for multivariate quadrature implementations.

**Trivariate Orthonormal Basis** Let us expand each probabilistic variable upto cubic terms ($d_1, d_2, d_3 = 3$), and choose Laguerre, Legendre and Hermite polynomials as corresponding univariate basis functions. Recall that $N_1 = d_1 + 1, N_2 = d_2 + 1$, and $N_3 = d_3 + 1$; which are the number of univariate basis set entries for each random variable. Thus, the orthonormal space for projection is constructed using tensor product with $N_1 = 4$, $N_2 = 4$ and $N_3 = 4$ functions in each variable, giving rise to 64 terms in the trivariate basis set. We have

$$\widehat{\psi}_l^y(y) = \begin{pmatrix} \widehat{L}_0^y(y_1)\widehat{P}_0^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_1^y(y_1)\widehat{P}_0^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_2^y(y_1)\widehat{P}_0^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_3^y(y_1)\widehat{P}_0^y(y_2)\widehat{H}_0^y(y_3) \\ \vdots \\ \widehat{L}_0^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_1^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_2^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_0^y(y_3) \\ \widehat{L}_3^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_0^y(y_3) \\ \vdots \\ \widehat{L}_0^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_3^y(y_3) \\ \widehat{L}_1^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_3^y(y_3) \\ \widehat{L}_2^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_3^y(y_3) \\ \widehat{L}_3^y(y_1)\widehat{P}_3^y(y_2)\widehat{H}_3^y(y_3) \end{pmatrix} \tag{8.34}$$

In compact form, we may write this as

$$\prod_{l=1}^{N} \widehat{\psi}_l^y(y) = \prod_{i=0}^{d_1}\prod_{j=0}^{d_2}\prod_{k=0}^{d_3} \widehat{L}_i^y(y_1)\widehat{P}_j^y(y_2)\widehat{H}_k^y(y_3) \tag{8.35}$$

**Trivariate Stochastic Projection:** The states and its time derivatives are

$$u(t,y) = \sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y),\ \ \dot{u}(t,y) = \sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y),\ \text{ and } \ddot{u}(t,y) = \sum_{i=1}^{N} \ddot{u}_i(t)\widehat{\psi}_i^y(y) \tag{8.36}$$

Using spectral expansions (8.36) in ODE (8.30) we get

$$\begin{gathered} m(y_1)\left(\sum_{i=1}^{N} \ddot{u}_i(t)\widehat{\psi}_i^y(y)\right) + c(y_2)\left(\sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y)\right) + k(y_3)\left(\sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y)\right) = 0 \\ \sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y) = u_0 \\ \sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y) = \dot{u}_0 \end{gathered} \tag{8.37}$$

Let us now project the above relations onto each basis element $\widehat{\psi}_j(y)$ using inner products that are numerically evaluated using quadrature rules. The stochastic ODE results from

$$\begin{gathered} \left\langle m(y_1)\sum_{i=1}^{N} \ddot{u}_i(t)\widehat{\psi}_i^y(y) + c(y_2)\sum_{i=1}^{N} \dot{u}_i(t)\widehat{\psi}_i^y(y) + k(y_3)\sum_{i=1}^{N} u_i(t)\widehat{\psi}_i^y(y) \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} \\ = \\ \left\langle 0 \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} \end{gathered} \tag{8.38}$$

with corresponding initial conditions as

$$\begin{gathered} \left\langle \sum_{i=1}^{N} u_i(0)\widehat{\psi}_i^y(y) \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} = \left\langle u_0 \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} \\ \left\langle \sum_{i=1}^{N} \dot{u}_i(0)\widehat{\psi}_i^y(y) \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} = \left\langle \dot{u}_0 \;\middle|\; \widehat{\psi}_j^y(y)\right\rangle_{\rho^y}^{\mathcal{Y}} \end{gathered} \tag{8.39}$$

#### 8.2.2.2 Probabilistic Moments

The orthonormal space for projection is constructed using tensor product with $N_1 = 4$, $N_2 = 4$ and $N_3 = 4$ functions in each variable, giving rise to 64 terms in the basis. The sparsity pattern arising due to this setup is shown in Figure 8.5. The mean and variance of the solution field is computed using sampling and projection methods are plotted in Figure 8.6. The SGM computations were performed using deterministic implementation of the SMD system and the system is solved for time interval of $[0, 10]s$ with a step size of $0.1s$ using BDF2 method. The stochastic collocation (sampling) solutions are computed using a tensor product grid of $15 \times 15 \times 15$. It can be seen that both the solutions are in good agreement with each other.

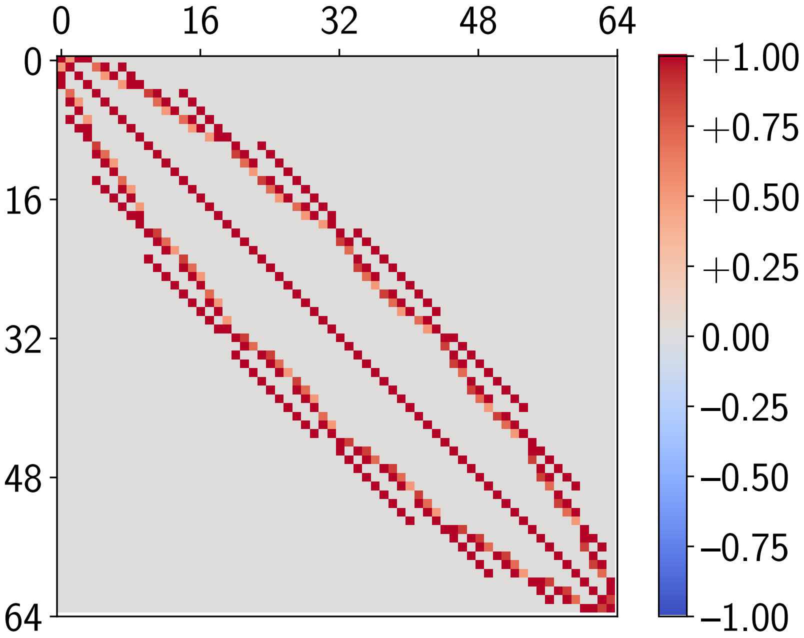
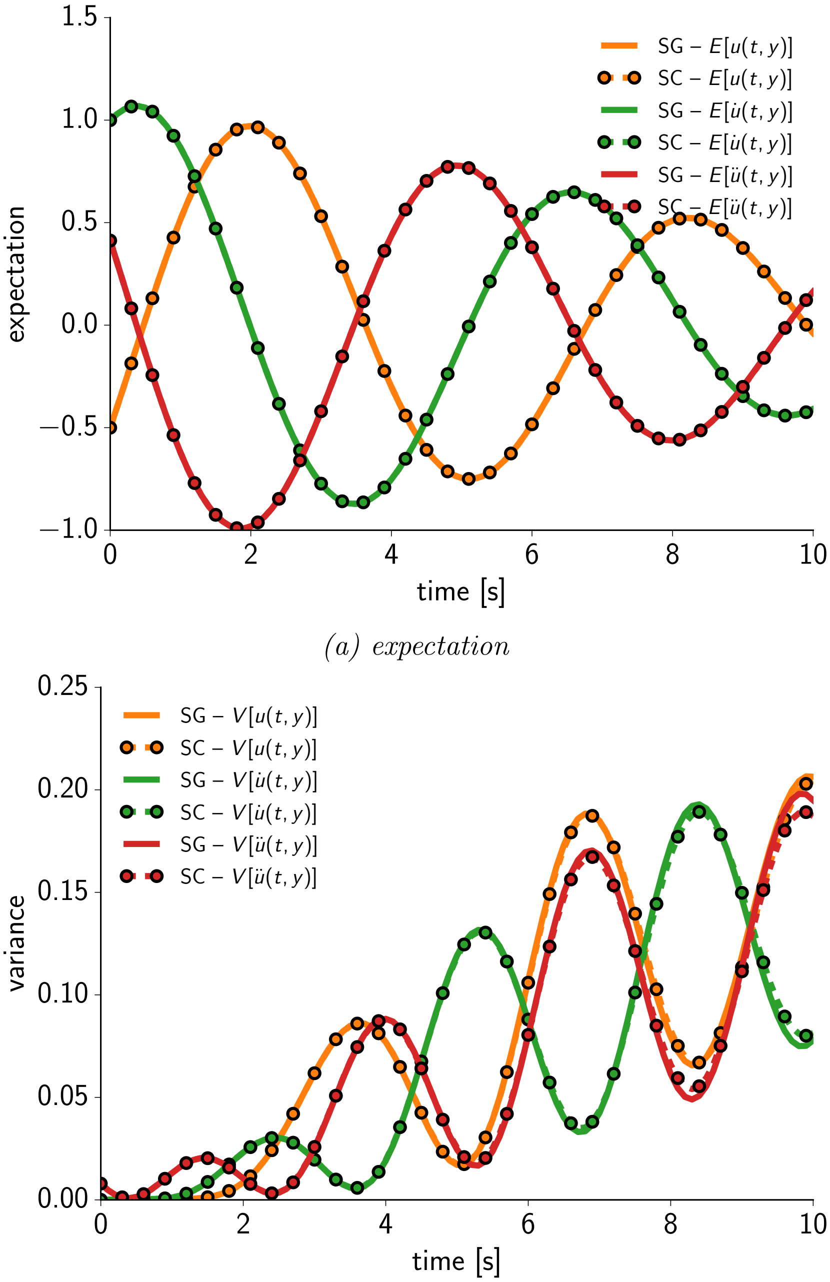


*Figure 8.5: Nonzero pattern of SMD system with 3 random variables $y_1$, $y_2$ and $y_3$ with $N_1 = N_2 = N_3 = 4$ giving rise to 64 basis terms with tensor product.*

#### 8.2.2.3 Function and Gradient Verification:

For the complex-step verification of adjoint gradients, we model the damping coefficient $c$ to be normally distributed as $\mathcal{N}(\mu = 0.2, \sigma = 0.1)$, and the mass $m$ is treated as the design



*(a) expectation*

*(b) variance*

*Figure 8.6: Expectation (top) and variance (bottom) of solution and its time derivatives obtained stochastic collocation and Galerkin methods.*

variable $\xi$. The stiffness value is assumed to be deterministic: $k = 5.0$. We are interested in computing the probabilistic moments of the time integral of potential energy

$$F = \int_0^T \frac{1}{2} k u(t)^2 \, dt \tag{8.40}$$

and the Kreisselmeier—-Steinhauser (KS) [45, 46] estimate of the maximum potential energy

$$F = a + \frac{1}{\rho_{ks}} \ln \left[ \int_0^T e^{\rho_{ks}\left(\frac{1}{2}ku(t)^2 - a\right)}\ dt \right]. \tag{8.41}$$

where $a$ and $\rho_{ks}$ are aggregation parameters. The probabilistic moments of (8.40) and (8.41) and their design variable derivatives computed using sampling and projection methods are listed in Tables 8.3 and 8.4, respectively. It be seen that the stochastic adjoint derivatives exhibit good agreement with the complex-step method.

*Table 8.3: Probabilistic moments and derivatives of the time integral of potential energy with 10 basis terms and 10 quadrature samples.*

| Quantity | Sampling | Projection |
|---|---|---|
| $\mathbb{E}[F]$ | $8.623478998405\mathbf{29308}$ | $8.623478998405\mathbf{32683}$ |
| $\mathbb{V}[F]$ | $2.18825865715\mathbf{865575}$ | $2.18825865715\mathbf{922419}$ |
| Adjoint $d\mathbb{E}[F]/dm$ | $2.29637589213\mathbf{232607}$ | $2.29637589213\mathbf{362459}$ |
| Complex-step $d\mathbb{E}[F]/dm$ | $2.296375892132\mathbf{85809}$ | $2.296375892132\mathbf{98554}$ |
| Error | $5.3 \times 10^{-13}$ | $6.4 \times 10^{-13}$ |
| Adjoint $d\mathbb{V}[F]/dm$ | $0.416863786\mathbf{527159164}$ | $0.416863786\mathbf{519840573}$ |
| Complex-step $d\mathbb{V}[F]/dm$ | $0.4168637865\mathbf{29946490}$ | $0.4168637865\mathbf{28287372}$ |
| Error | $2.8 \times 10^{-12}$ | $8.4 \times 10^{-12}$ |

*Table 8.4: Probabilistic moments and derivatives of the maximum potential energy in time domain with 10 basis terms and 10 quadrature samples.*

| Quantity | Sampling | Projection |
|---|---|---|
| $\mathbb{E}[F]$ | $2.50\mathbf{2931279813640 05}$ | $2.50\mathbf{708458655184918}$ |
| $\mathbb{V}[F]$ | $3.8\mathbf{1254813515422378} \times 10^{-3}$ | $3.8\mathbf{0147552479659367} \times 10^{-3}$ |
| Adjoint $d\mathbb{E}[F]/dm$ | $-3.69267283814\mathbf{737500} \times 10^{-3}$ | $-3.69267283814\mathbf{618151} \times 10^{-3}$ |
| Complex-step $d\mathbb{E}[F]/dm$ | $-3.692672838150\mathbf{61589} \times 10^{-3}$ | $-3.692672838150\mathbf{30625} \times 10^{-3}$ |
| Error | $3.2 \times 10^{-15}$ | $2.5 \times 10^{-14}$ |
| Adjoint $d\mathbb{V}[F]/dm$ | $-8.59422933369\mathbf{482361} \times 10^{-3}$ | $-8.5942293336\mathbf{8299627} \times 10^{-3}$ |
| Complex-step $d\mathbb{V}[F]/dm$ | $-8.5942293337 0\mathbf{226558} \times 10^{-3}$ | $-8.59422933370\mathbf{156128} \times 10^{-3}$ |
| Error | $7.4 \times 10^{-15}$ | $8.1 \times 10^{-14}$ |

### 8.2.3 Natural Vibration of a Series of Masses and Springs

We consider a linear ordinary differential equation modeling a series of masses connected through springs shown in Figure 8.7. The residual of the equations of motion are of the abstract form $R(t, \ddot{u}(t), \dot{u}(t), u(t)) = 0$, and can be written out as

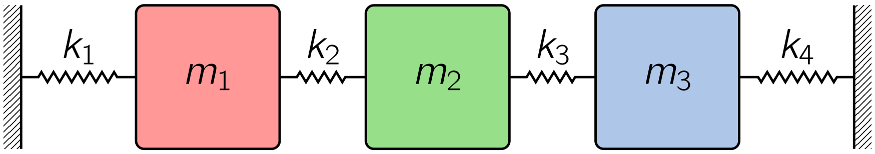


*Figure 8.7: The mass spring system.*

$$\begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \begin{Bmatrix} \ddot{u}_1 \\ \ddot{u}_2 \\ \ddot{u}_3 \end{Bmatrix} + \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 + k_4 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = 0 \qquad (8.42)$$

with initial conditions $u = [0, 0, 0]$ and $\dot{u} = [0, 0, 0.1]$, as well as spring stiffness constants $k_1 = 1,\ k_2 = 10,\ k_3 = 100$ and $k_4 = 1000$. In order to solve the solve second-order dynamic system "as-is" in natural form [119, 120], we implement the residual evaluation based on Equation (8.42), and corresponding Jacobian matrix blocks

$$\frac{\partial R}{\partial \ddot{u}} = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \text{ and } \frac{\partial R}{\partial u} = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 + k_4 \end{bmatrix}$$

within the TACS finite element framework, following the `Element` interface outlined in Section 7.3. The temporal solution of the system is setup using the second-order backward differences method for a finite time interval of $[0, 2]s$ with a step size of $0.02s$.

**Probabilistic Parameters, Quadrature and Basis Setup:** For the extension of the above deterministic analysis to stochastic Galerkin computations, we assume the masses are dependent on probabilistically modeled random variables as $m_1 = m_1(y_1)$, $m_2 = m_2(y_2)$, $m_3 = m_3(y_3)$ where

1. $y_1 \sim \mathcal{U}(a = 1.0, b = 2.0)$ follows uniform distribution,
2. $y_2 \sim \mathcal{E}(\mu = 10.0, \beta = 1.0)$ follows exponential distribution,
3. $y_3 \sim \mathcal{N}(\mu = 100.0, \sigma = 10.0)$ follows normal distribution.

Thus, we have a stochastic ODE in $\mathcal{T} \otimes \mathcal{Y}^3$ whose stochastic states are $u(t, y)$, $\dot{u}(t, y)$ and $\ddot{u}(t, y)$. The numerical quadrature in three probabilistic dimensions is setup using the tensor product of Gauss–Legendre, Gauss–Laguerre and Gauss-Hermite quadrature points and weights along each of the three probabilistic dimensions. Next, we select univariate Legendre, Laguerre and Hermite orthonormal polynomials listed in Table 6.1 of chosen maximum degrees $[d_2, d_2, d_3]$. Using these we construct a set of distinct trivariate orthonormal polynomials using tensor product rule and complete polynomial rule. We use this mass–spring system to compare the efficiency of these two basis choices.

**Semi-Intrusive Projection of Arrays and Matrices:** With the setup of probabilistic basis and quadrature, the stochastic states follow from (7.11), the stochastic residual follows from (7.12), the stochastic Jacobian follows from (7.14), and initial conditions follow from (7.36). For this simple system, the Jacobian matrix is independent of the state variables and thus are easier to compute. The stochastic mass matrix entries are computed as $\sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i(y_q) \widehat{\psi}_j(y_q) [\frac{\partial R}{\partial \ddot{u}}(y_q)]$. Note that the problem size is $3N$, where $N$ is the number of terms in the trivariate basis set.

**Probabilistic Moments:** We compare the probabilistic moments computed using the semi-intrusive stochastic Galerkin method with that of stochastic sampling. The reference solution is computed using the stochastic sampling method with a grid of $10 \times 10 \times 10$ amounting to 1000 quadrature samples. The mean and variance of the solution field is computed using sampling and projection methods are plotted in Figure 8.8 along with a band whose width is one standard deviation. Here we compare semi-intrusive SGM with 20 basis terms formed using a maximum degree set of $[3, 3, 3]$ parameter wise with the sampling-based solution. It can be seen that both the solutions are in good agreement with each other. Next, we compare the norm of the absolute error, defined as the difference between sampling- and projection-based solutions in Figure 8.9. For this study we use both tensor and complete polynomial based construction of basis function set. It can be seen that the moments (expectation and variance) computed using polynomial based construction is accurate than tensor product construction.

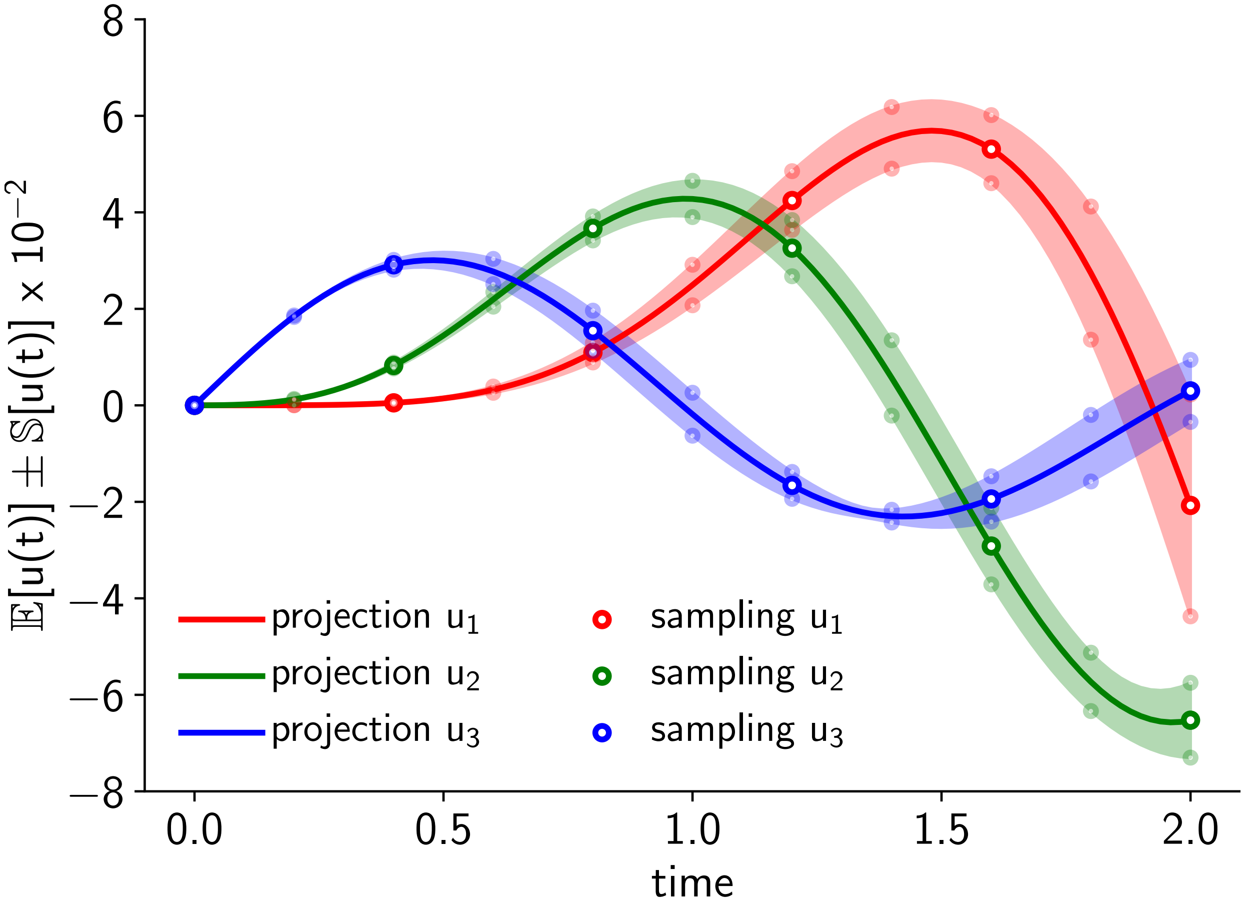


*Figure 8.8: The time history of the expected response along with a band that is one standard deviation wide on either side.*

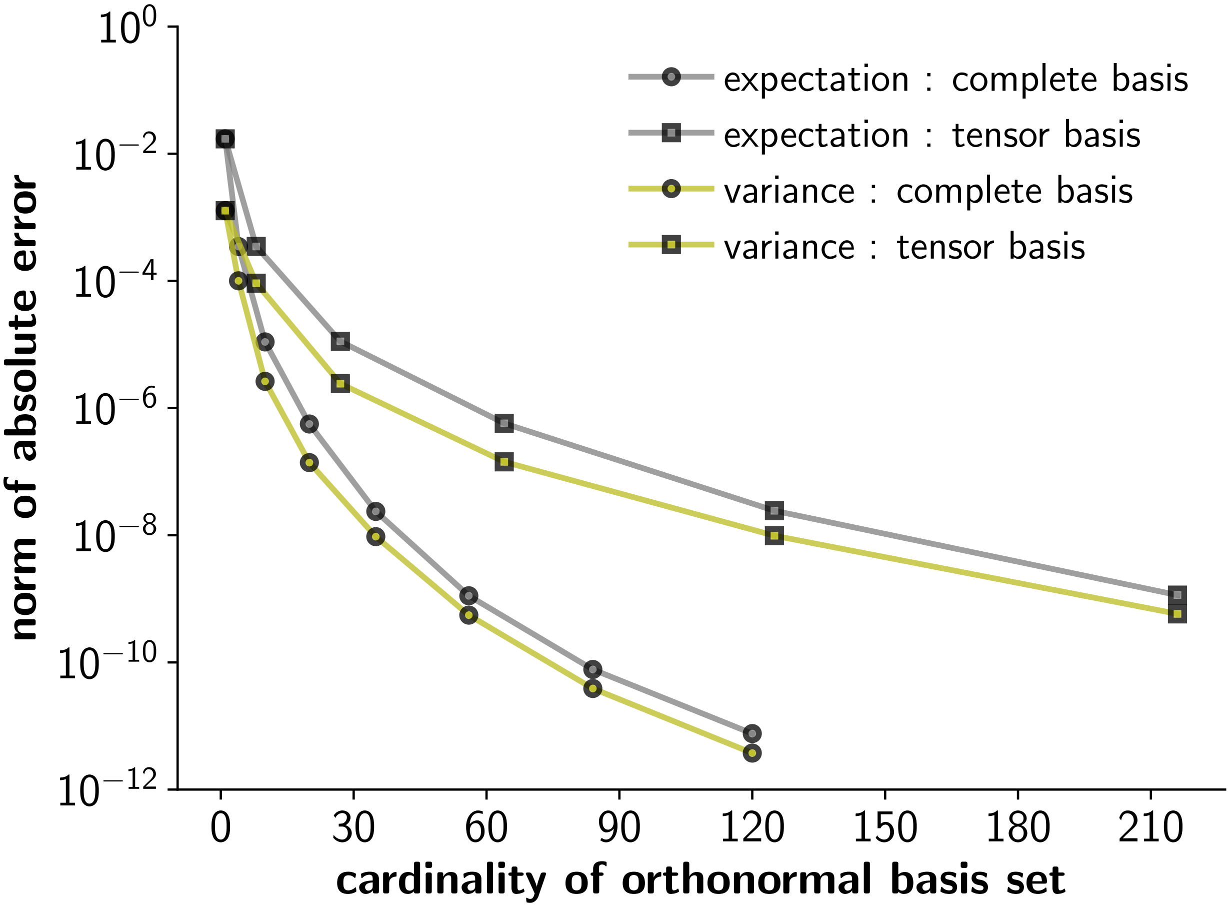


*Figure 8.9: Plot of the norm of absolute error in expectation and variance versus the number of terms in the basis set for different basis sets.*

### 8.2.4 Pitching and Plunging Airfoil System

We study a second-order differential equation in two variables: pitch and plunge variables of an airfoil with prescribed initial conditions $u_0$, $\dot{u}_0$ and constant system parameters $[M]$, $[C]$ and $[K]$ in matrix form

$$\begin{aligned}
\begin{bmatrix} m & s \\ s & I_f \end{bmatrix} \begin{Bmatrix} \ddot{u}_1(t) \\ \ddot{u}_2(t) \end{Bmatrix} + \begin{bmatrix} c_h & 0 \\ 0 & c_a \end{bmatrix} \begin{Bmatrix} \dot{u}_1(t) \\ \dot{u}_2(t) \end{Bmatrix} + \begin{bmatrix} k_h & 0 \\ 0 & k_a \end{bmatrix} \begin{Bmatrix} u_1(t) \\ u_2(t) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \quad &\in \quad \mathcal{T} \\
u(0) = u_0 \quad &\in \quad \partial\mathcal{T} \\
\dot{u}(0) = \dot{u}_0 \quad &\in \quad \partial\mathcal{T}
\end{aligned} \tag{8.43}$$

The deterministic parameters of the system are listed in Table 8.5. Let us assume that

*Table 8.5: Parameters defining the pitching and plunging airfoil system.*

| Parameter | Definition | Value | Unit |
|---|---|---|---|
| $x_f$ | position of flexural axis | 0.25 | $m$ |
| $x_{cm}$ | position of center of mass | 0.375 | $m$ |
| $m$ | mass of airfoil | 55.3291 | $kg$ |
| $I_f$ | mass moment of inertia of the airfoil around the elastic axis | 3.4581 | $kg.m^2$ |
| $s$ | static unbalance $m(x_{cm} - x_f)$ | 6.9161375 | $kg.m$ |
| $c_h$ | plunge damping | 0 | $N/kg/s$ |
| $c_a$ | pitch torsional damping | 0 | $N.m/kg/s$ |
| $k_h$ | plunge stiffness | 11366.0 | $N/kg$ |
| $k_a$ | pitch torsional stiffness | 7002.6 | $N.m/kg$ |

the mass $m := m(y)$, where $y$ is a random variable from stochastic domain $\mathcal{Y}$. This adds a stochastic dimension to the differential equation resulting in the stochastic differential equation. In $\mathcal{T} \otimes \mathcal{Y}$ we have the vector-valued ordinary differential equation

$$\begin{bmatrix} m(y) & s(y) \\ s(y) & I_f \end{bmatrix} \begin{Bmatrix} \ddot{u_1}(t,y) \\ \ddot{u_2}(t,y) \end{Bmatrix} + \begin{bmatrix} c_h & 0 \\ 0 & c_a \end{bmatrix} \begin{Bmatrix} \dot{u_1}(t,y) \\ \dot{u_2}(t,y) \end{Bmatrix} + \begin{bmatrix} k_h & 0 \\ 0 & k_a \end{bmatrix} \begin{Bmatrix} u_1(t,y) \\ u_2(t,y) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \tag{8.44}$$

In $\partial\mathcal{T} \otimes \mathcal{Y}$ we have initial conditions defined as

$$\begin{aligned} u(0,y) &= u_0 \\ \dot{u}(0,y) &= \dot{u}_0. \end{aligned} \tag{8.45}$$

When using the projection method, an extended linear system is formed. The sparsity of corresponding the stochastic Jacobian for pitching-plunging airfoil system can be visualized from Figure 8.10. We find the moments of the solution using SGM and SCM and compare them in Figure 8.11. It can be observed that both the methods are in excellent agreement for both the degrees of freedom (pitch and plunge) for both statistical moments (mean and variance).

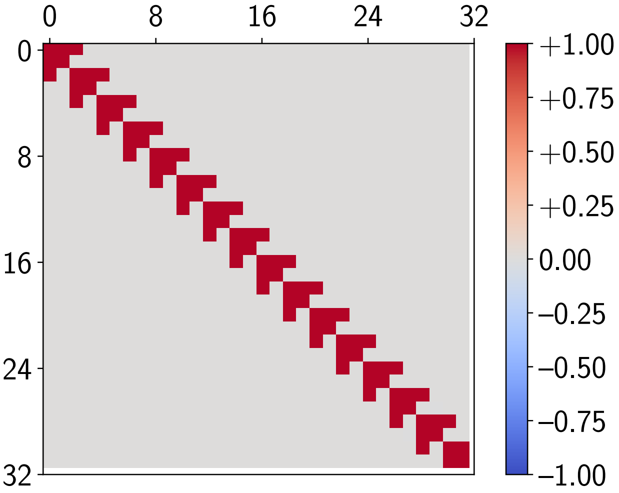


*Figure 8.10: Nonzero pattern of PPA system with one random variable decomposed on a stochastic basis with 16 terms.*

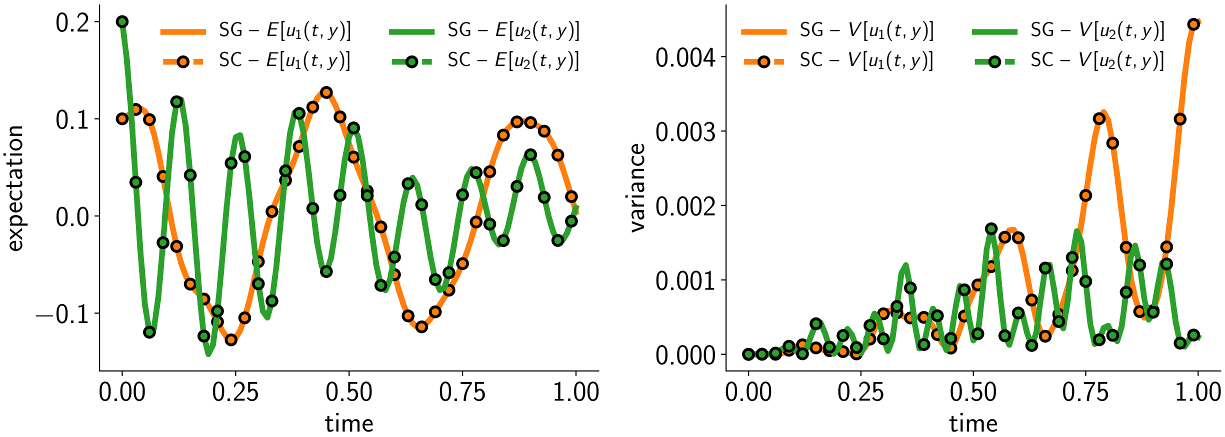


*Figure 8.11: Expectation (left) and variance (right) of solution of pitching-plunging airfoil system obtained using stochastic Galerkin with 5 terms in the basis set and collocation methods with 15 samples.*

## 8.3 Nonlinear Time Dependent Systems

### 8.3.1 Van der Pol Oscillator

The natural form governing differential equation for Van der Pol oscillator is

$$R(t, \ddot{u}(t), \dot{u}(t), u(t)) := \ddot{u}(t) - \mu(1 - u(t)^2) \cdot \dot{u}(t) + u(t) = 0 \tag{8.46}$$

which is a scalar nonlinear second-order differential equation where $u(t), \dot{u}(t)$ and $\ddot{u}(t)$ are unknown scalar functions of independent parameter time $t$. Let the initial conditions be $u(0) = 1$ and $\dot{u}(0) = 1$. The partial Jacobians required for linearization of (8.46) are

$$\frac{\partial R}{\partial \ddot{u}} = 1, \quad \frac{\partial R}{\partial \dot{u}} = -\mu\left(1 - u(t)^2\right) \quad \text{and} \quad \frac{\partial R}{\partial u} = 1 + 2\mu u(t)\dot{u}(t). \tag{8.47}$$

We consider the case where the oscillator parameter $\mu$ is a function of normally distributed random variable $y \sim \mathcal{N}(1.0, 0.25)$. The stochastic nonlinear ODE is

$$\mathcal{R}(t, y, \ddot{u}(t, y), \dot{u}(t, y), u(t, y)) := \ddot{u}(t, y) - \mu(1 - u(t, y)^2) \cdot \dot{u}(t, y) + u(t, y) = 0 \tag{8.48}$$

The solution of the stochastic ODE using semi-intrusive projection is centered around implicitly forming the stochastic residuals, Jacobians and initial conditions as described in Section 7.2. The probabilistic moments determined using semi-intrusive projection are shown in Figures 8.12 and  8.13.

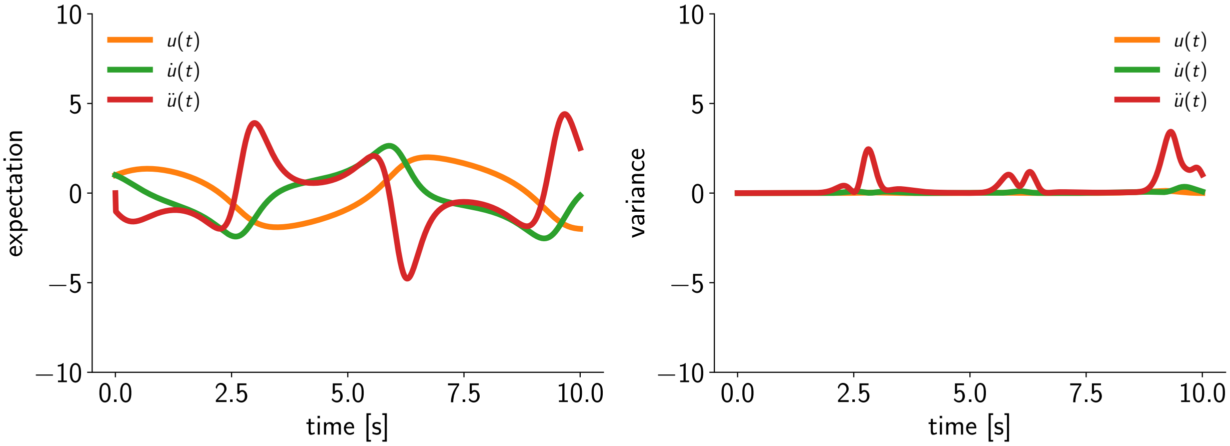


*Figure 8.12: Expectation (left) and variance (right) of solution of Van der Pol oscillator obtained using projection with 7 terms.*

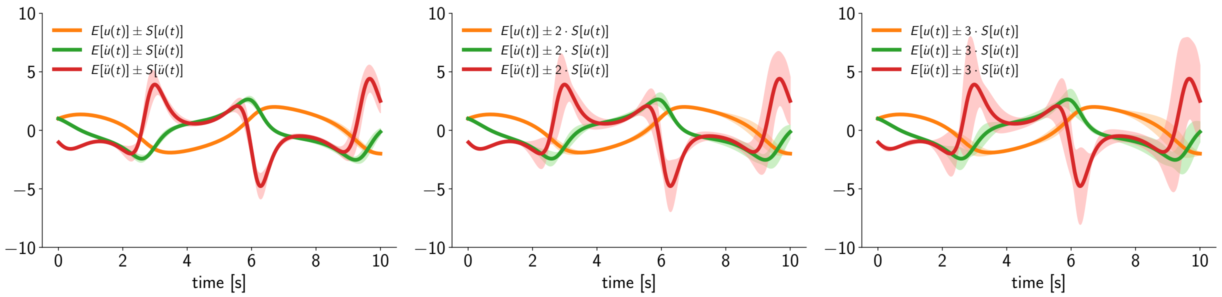


*Figure 8.13: The expected response quantities with overlaid bands of one (left), two (middle) and three (right) standard deviations using projection with 7 terms.*

## 8.4 Finite Element Based Flexible Multibody Systems

We present the application of semi-intrusive SGM on finite element problems in the context of flexible multibody dynamics.

### 8.4.1 Four-Bar Mechanism

We present the application of the proposed semi-intrusive projection technique to the four-bar mechanism benchmark case [1].

#### *8.4.1.1 Analysis Setup*

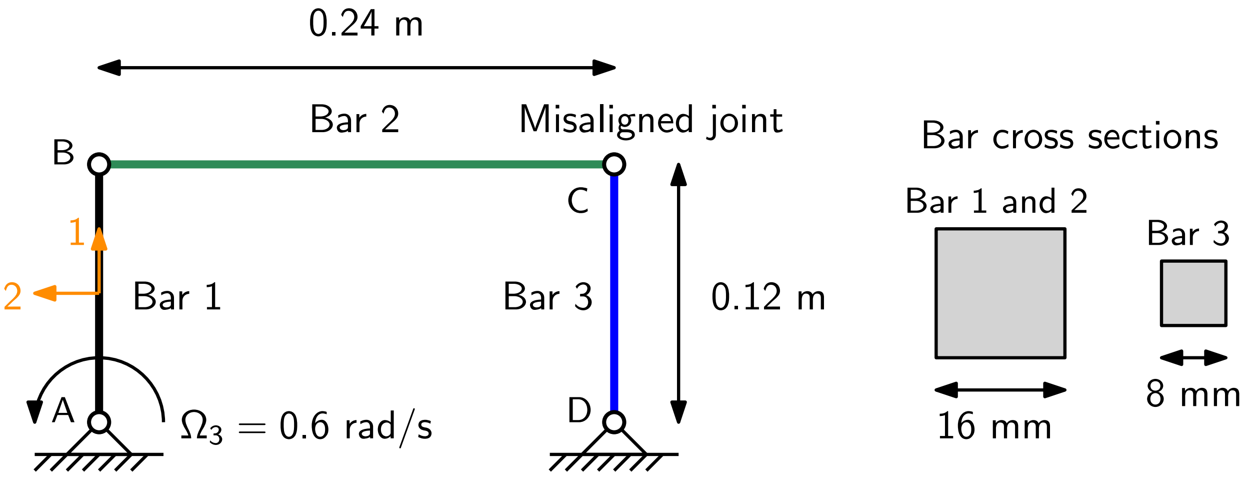


*Figure 8.14: The four-bar mechanism problem.*

Figure 8.14 illustrates the setup of the four-bar mechanism. The problem contains three flexible bars that are modeled using Timoshenko beam elements, three revolute joints and an actuator driving the mechanism. An imaginary, infinitely rigid fourth bar exists in the mechanism between the points $A$ and $D$. The revolute joints are at points $A$, $B$, and $D$, and have an axis of rotation that is perpendicular to the plane of the mechanism. The revolute joint at point $C$ is misaligned by an angle of 5°, and rotated about the direction of the bar $CD$. This misalignment angle is modeled as subject to uncertainty and distributed normally with $\mathcal{N}(\mu = 5°, \sigma = 2.5°)$. The Bars $AB$ and $BC$ are of the same cross-section, while bar $CD$ has a smaller cross-section. The rotation of bar $AB$ about point $A$ of the mechanism is driven at an angular rate of $\Omega_3 = 0.6\,\mathrm{rad/s}$. The material properties are Young's modulus of 207 $GPa$, density of 7800 $kg/m^3$, Poisson's ratio 0.3, and shear correction factor of 5.0/6.0. The angular rate of the revolute driver is 0.6 $rad/s$, due to which it takes 12 $s$ for one full revolution of the mechanism. The time marching is performed using second-order BDF method [120]. The finite element library TACS is used for deterministic, stochastic sampling and stochastic projection based solution of the four-bar mechanism problem.

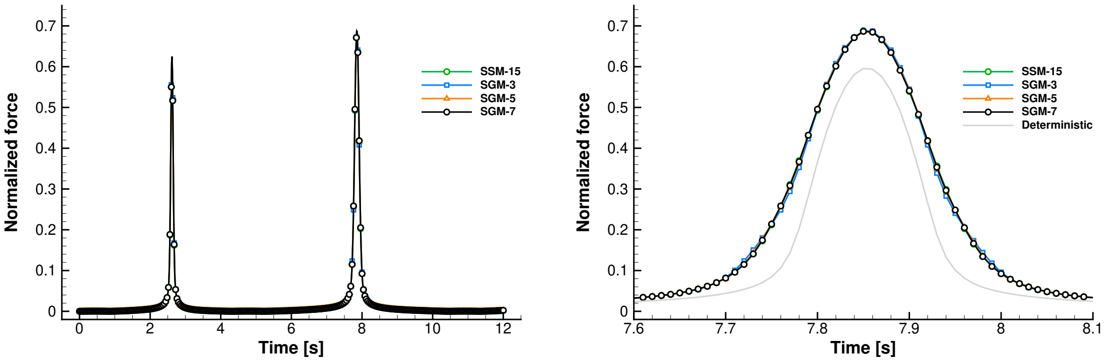


*Figure 8.15: The mean of normalized axial force in bar AB as a function of time predicted using SGM and SSM.*

#### *8.4.1.2 Verification of Probabilistic Moments*

Our goal here is to demonstrate the accuracy of the semi-intrusive stochastic Galerkin method by comparing with the stochastic sampling method. Figure 8.15 shows the response of the expectation of the normal force in the bar $AB$ computed using semi-intrusive SGM with 3,

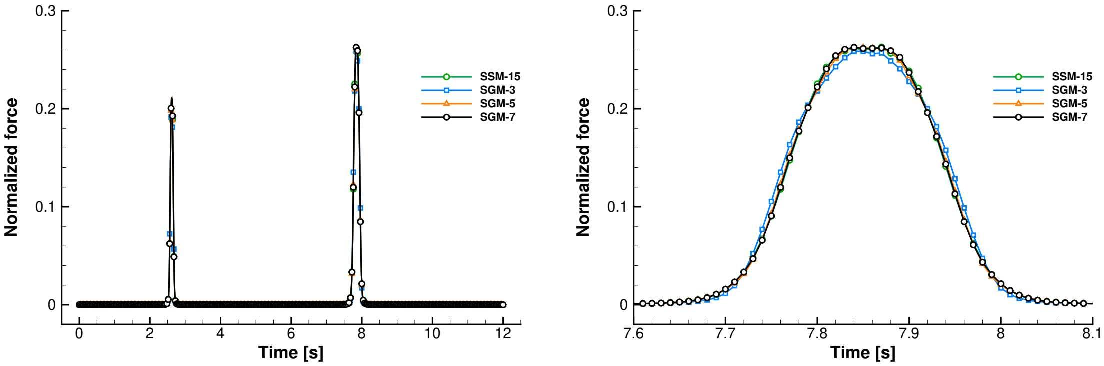


*Figure 8.16: The variance of the normalized axial force in bar AB as a function of time predicted using SGM and SSM.*

5, and 7 terms in the orthonormal basis compared with SSM using 15 sample points. In one full cycle between 0 to 12 $s$, the normal force exhibits two larger peaks that occur as the mechanism is forced to snap through the angle where it would lock if the bars were rigid due to the misaligned joint. The overall behavior of the mean axial force is shown in Figure 8.15 as well as a zoomed in view of the behavior between $t = 7.6$ and $t = 8.1$ which centers on the second large spike in the axial force. The SGM captures the peak behavior in the normal force, even with only three terms. In the case of the deterministic simulation without uncertainties (represented in gray), we obtain a normal force that is less than the mean maximum force considering uncertainties. Figure 8.16 shows the variance of the axial force in the bar $AB$ computed using SGM with 3, 5, and 7 terms in the orthonormal basis compared with SSM using 15 sample points. Again, the distribution of the variance exhibits two large peaks. The second zoomed in view of the variance illustrates that SGM again captures the overall behavior with only 3 terms. In general a better agreement is obtained between SSM and SGM as more basis functions are used.

#### *8.4.1.3 Optimization Under Uncertainty*

Next, we extend the analysis case presented to optimization demonstration. We also verify the accuracy of the adjoint-gradients of expectation and variance using the complex-step method prior to optimization.

**(a) Optimization Setup:** The optimization under uncertainty problem is stated as:

| | |
|---|---|
| **minimize** | $\mathbb{E}[\text{mass}]$ |
| **design variable** | width of bars |
| **uncertainty** | revolute axis $\theta \sim \mathcal{N}(\mu = 5^\circ, \sigma = 2.5^\circ)$ |
| **subject to** | $\mathbb{E}[\text{failure}] + \beta \cdot \mathbb{S}[\text{failure}] \leq 1$ |
| | $\mathbb{E}[\text{displacement}] + \beta \cdot \mathbb{S}[\text{displacement}] \leq 5mm$ |
| **bounds** | $5mm \leq \text{width} \leq 25mm$ |

The mass objective refers to the overall mass of the mechanism, the displacement constraint refers to the displacement component that is out of the plane, and the failure is evaluated based on allowed normal (axial) force in the bars. We use spatio-temporal aggregation of constraint functions based on the Kreisselmeier–Steinhauser formulation [45, 46] for displacement and failure.

**(b) Gradient Verification:** The adjoint gradients of the expectation and variance are verified using the complex-step method, and the values are compared in Table 8.6. It can be seen that the values are in good agreement and show the consistency of adjoint implementation with that of the complex perturbation. Since, we implicitly formed the adjoint equations, this shows that the approach is equivalent to the approach where one derives explicit adjoint equations.

**(c) Optimization Results:** We perform five optimization runs composed of one deterministic and four probabilistic OUU runs with $\beta = 0, 1, 2$ and 3, and compare the designs in Table 8.7. As a general remark, the designs are heavier with more and more incorporation of probabilistic criteria in the formulation. The widths of the bars AB and BC have a larger impact than the width of the third bar CD, as the bar BC encounters the highest magnitude of force and displacement throughout the simulation range.

*Table 8.6: The complex-step verification of adjoint derivatives of expectation and variance of objective and constraint metrics.*

| Quantity | Mass | Failure | Displacement |
|---|---|---|---|
| adjoint $d\mathbb{E}[F]/d\xi$ | 1078.272 | 22.5748 | -0.1067834 |
| complex $d\mathbb{E}[F]/d\xi$ | 1078.272 | 22.5748 | -0.1067834 |
| error | $4.5 \cdot 10^{-11}$ | $8.7 \cdot 10^{-6}$ | $5.5 \cdot 10^{-8}$ |
| adjoint $d\mathbb{V}[F]/d\xi$ | N/A | 22.45792 | $-1.57732 \cdot 10^{-4}$ |
| complex $d\mathbb{V}[F]/d\xi$ | N/A | 22.45792 | $-1.57732 \cdot 10^{-4}$ |
| error | N/A | $6.7 \cdot 10^{-6}$ | $1.1 \cdot 10^{-10}$ |

*Table 8.7: Designs resulting from the deterministic and probabilistic optimization of the four-bar mechanism.*

| Quantity | Deterministic | $\beta = 0$ | $\beta = 1$ | $\beta = 2$ | $\beta = 3$ |
|---|---|---|---|---|---|
| width AB & BC | 5.0 | 5.0 | 13.0 | 18.3 | 23.5 |
| width CD | 5.0 | 5.0 | 5.0 | 6.0 | 7.0 |
| mass [kg] | 1.1 | 1.1 | 6.02 | 11.7 | 19.3 |
| failure [% max] | 47% | 55 | 90 | 100 | 100 |
| displacement [% max] | 78.22 | 78.72 | 100 | 100 | 100 |

### 8.4.2 Flexible Remote Manipulator System (Canadarm)

Next, we apply the semi-intrusive stochastic Galerkin method for the probabilistic design of a flexible robotic manipulator system, that is representative of the robotic arms used in space, such as the Canadarm-I, the Canadarm-II and the Dextre. These flexible manipulator systems [122–128] are used to move payloads in space, assemble space systems, assist with the docking of space shuttles from earth, and perform maintenance activities in space (see Figure 8.17).

#### *8.4.2.1 Analysis Setup*

The representative system used here is modeled after the Canadarm-I, and is functionally similar to a human arm with six joint degrees of freedom. Figure 8.18 shows the schematic of the manipulator system modeled using the TACS flexible multibody dynamics framework.

*Figure 8.17: The working of shuttle manipulator systems* [a].

There are two joints at the shoulder end, one joint at the elbow between flexible booms and three joints at the wrist end of the robotic manipulator system. The joints at $A$ and $E$ allow yawing motion, the joints at $B$, $C$ and $D$ allow pitching motion, and the joint at $F$ allows rolling motion. The booms in Canadarm-I are made of graphite epoxy, but for our purposes here, we assume the material properties are Young's modulus of 207 $GPa$, density of 7800 $kg/m^3$, Poisson's ratio 0.3, and shear correction factor of 5.0/6.0. We use ten Timoshenko beam elements for each boom with rectangular cross-sections for the finite element analysis. The number of degrees of freedom in the problem is 432. The angular rate of the joints are assumed to be $\omega_A = 0.1\ rad/s$, $\omega_B = 0.1\ rad/s$, $\omega_C = 0.1\ rad/s$, $\omega_D = 0.1\ rad/s$, $\omega_E = 0.1\ rad/s$, and $\omega_F = 0.1\ rad/s$ about their respective revolute axes. The lengths are $l_1 = 0.9\ m,\ l_2 = 6.4\ m,\ l_3 = 7.0\ m,\ l_4 = 0.5\ m,\ l_5 = 0.8\ m,\ l_6 = 0.6\ m$. The masses of rigid bodies are $m_1 = 95\ kg,\ m_4 = 8\ kg,\ m_5 = 44\ kg,\ m_6 = 41\ kg$ [122–124] and

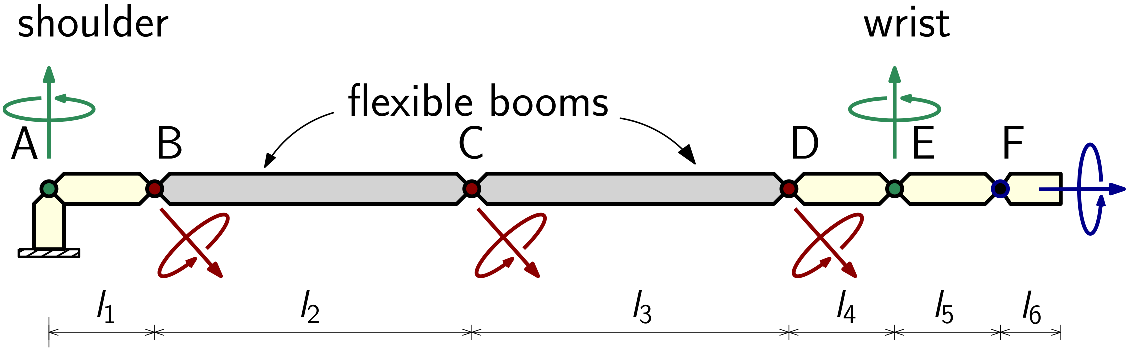


*Figure 8.18: A six degree of freedom remote manipulator system.*

[a] https://www.nasa.gov/mission_pages/shuttle/behindscenes/rms_gallery.html

the payload mass is $100,000\ kg$. The dynamics of the robotic manipulator is simulated for a duration of $5s$ with BDF2 time marching scheme. The time lapse of the simulated motion is shown in Figure 8.19, with aforementioned angular rates and initial configuration of a fully extended arm.

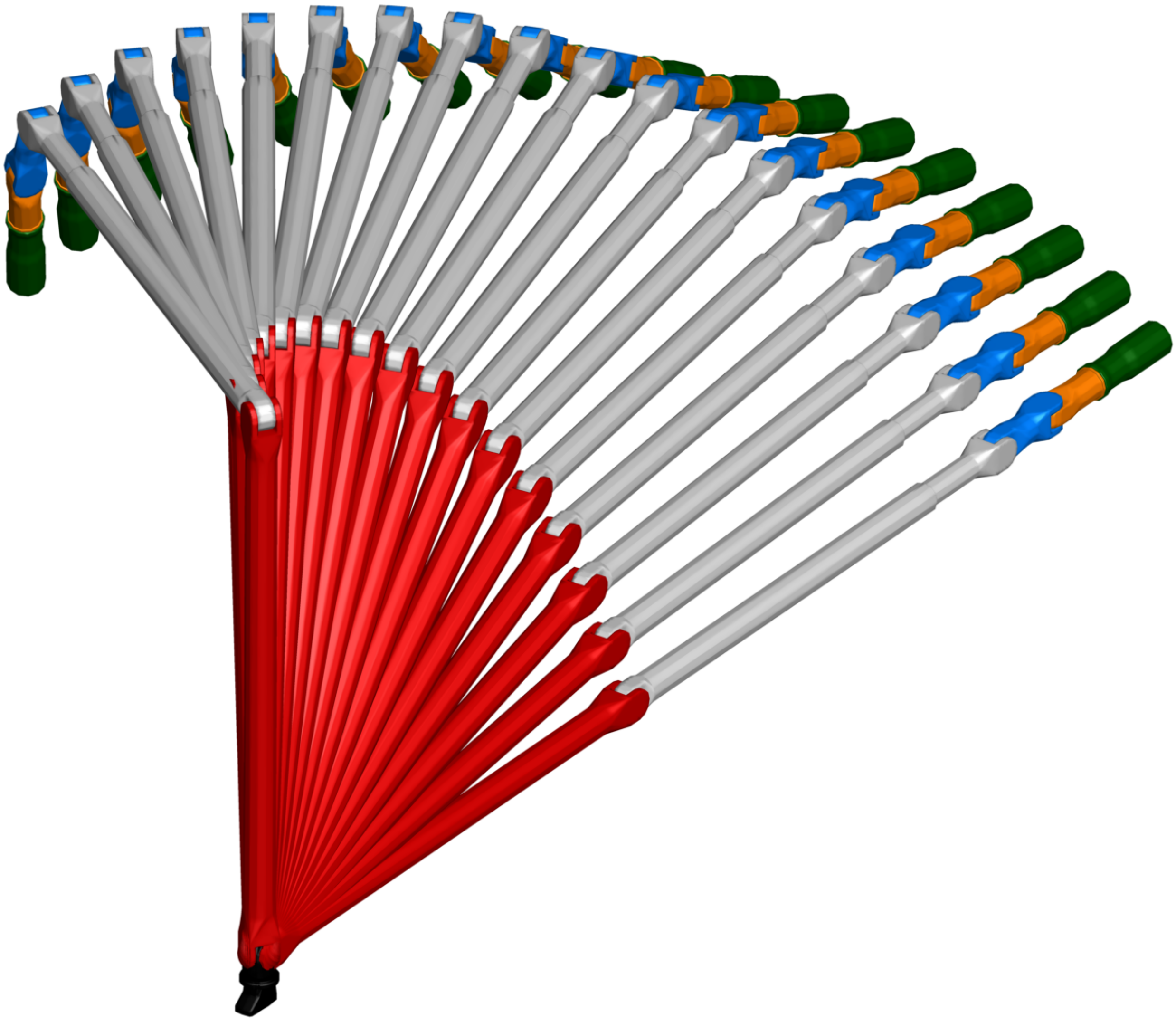


*Figure 8.19: Timelapse of the motion of Canadarm model.*

**Remark on configurations and rates:** The intended use of open-chain mechanisms like the Canadarm is to reach the three dimensional space within the full extent of the mechanism, often, within a specified amount of time.

1. This implies that there are an *infinite number of attainable configurations* to be analyzed (even by excluding elastic deformations), which is computationally intractable. However, a finite subset of preferred kinematic configurations can be simulated and analyzed with the availability of computing power. One possible application of UQ methods here is that, the initial configuration of the bodies can be modeled probabilistically and defined as random functions following specified probability distributions.

2. Secondly, there are an *infinite number of rates* at which the configurations can be attained. The rates at which the links are driven is often subject to upper bounds for safety reasons: for example, the tip of the Canadarm-I is designed to operate at a maximum rate of 0.06 $m/s$ [1]. In order to mathematically model such uncertainties during operation, we can use the probabilistic random variables $y$ with appropriate distributions, instead of analyzing different possible rates.

In summary, the approach of using probabilistically modeled initial conditions along with probabilistically modeled joint angular rates, can provide a firm mathematical basis for analysis and optimization. However, proper care must be exercised in "assuming" distributions for such parameters, if needed, the inverse UQ methods should be applied.

#### *8.4.2.2 Optimization Under Uncertain Payloads*

Next, we demonstrate the utility of OUU as a tool to account for uncertainties in space-systems design process with a simplified probabilistic model of the manipulator system. The Canadarm is used to move bodies whose mass ranges from a few hundred kilograms (*e.g.* astronauts on spacewalk wearing specialized suits) to several hundred tonnes (*e.g.* assembling and repairing the space station). The robotic manipulator system needs to be designed for handling a wide range of masses, and we demonstrate the suitability of the OUU methods in this context. We model the mass of the payload to depend on a "probabilistic" random variable – to emulate the scenario where the arm is used to move payloads of different masses

[1] https://www.ieee.ca/millennium/canadarm/canadarm_technical.html

around space, that is $mass_{payload}(y)$ where $y \sim \mathcal{N}(\mu = 100,000\ kg, \sigma = 50,000\ kg)$. We setup the optimization problem to minimize the mass of the system subject to stress-based failure constraint as follows:

$$\begin{array}{ll} \textbf{minimize} & \mathbb{E}[\text{mass}] = \mathbb{E}[\rho(A_1 l_1 + A_2 l_2)] = \rho(A_1 l_1 + A_2 l_2) \\ \textbf{design variable} & \text{width of bars} \\ \textbf{uncertainty} & \text{payload mass} \sim \mathcal{N}(\mu = 100,000\ kg, \sigma = 50,000\ kg, N = 3) \\ \textbf{subject to} & \mathbb{E}[\text{failure}] + \beta \cdot \mathbb{S}[\text{failure}] \leq 1 \\ \textbf{bounds} & 25cm \leq \text{width} \leq 50cm \end{array} \tag{8.49}$$

The design variables are the cross-sectional width of the booms. The objective function refers to the mass of the flexible booms subject to design, where $\rho$ is the density of the material, $A_i$ is the area of cross-section, and $l_i$ is the length of the $i-$th boom. In this case, the objective function has no dependence on the random variable $y$; but for verification purposes we evaluate the expectation and variance operators during computations. We use spatio-temporal aggregation of constraint functions based on the Kreisselmeier–Steinhauser formulation [45, 46] for failure constraint evaluation. We use a probabilistic basis set with $N = 3$ Hermite polynomials. This implies that the stochastic matrices and vectors are thrice as big as the deterministic counterparts, and are formed implicitly using the semi-intrusive stochastic Galerkin method.

**(a) Gradient Verification:** First, the verification of derivatives is performed using the complex-step method prior to optimization, and the values are listed in Table 8.8. We see a good agreement in derivative values for optimization; however, we note that the accuracy of the adjoint derivatives is affected by a few significant digits when spatio-temporal aggregation is employed. We believe that this is due to the numerical issues arising from the choice of aggregation parameter, which needs to be large number $>> 1$ for better approximation of the maximum.

*Table 8.8: The complex-step verification of adjoint derivatives for the Canadarm system.*

| Quantity | Mass | Failure |
|---|---|---|
| adjoint $d\mathbb{E}[F]/d\xi$ | $1.2480000000000\underline{1979} \cdot 10^5$ | $\text{-}3.7659\underline{7889920338691}$ |
| complex $d\mathbb{E}[F]/d\xi$ | $1.2480000000000\underline{1819} \cdot 10^5$ | $\text{-}3.7659\underline{6706242138746}$ |
| relative error | $1.3 \cdot 10^{-15}$ | $3.1 \cdot 10^{-6}$ |
| adjoint $d\mathbb{V}[F]/d\xi$ | N/A | $-4.46442271585651973 \cdot 10^{-1}$ |
| complex $d\mathbb{V}[F]/d\xi$ | N/A | $-4.46444953483493667 \cdot 10^{-1}$ |
| relative error | N/A | $6.0 \cdot 10^{-6}$ |

**(b) Optimization Results:** The optimization problem (8.49) was solved for reliability parameter values ranging from zero to seven. For the purposes of comparison, a deterministic optimization problem with reference payload mass of $100,000$ *kg* was also solved. The results are tabulated in Table 8.9. It can be seen that the widths increase as we require more constraint reliability through the parameter $\beta$. It appears that the constraints are 100% active for some OUU designs (*i.e.* $\beta = 3 - 7$), but recall that the mathematical constraint formulation included $\beta$ standard deviations of failure into consideration. To substantiate this further, the expected constraint manifold can be seen to be more and more away from the actual enforced constraint from the tabulated "expectation" values. Therefore, the reliability parameter is similar in purpose to the *factor of safety* commonly employed in structural design, as they both seek a design point that is a "specified" distance away from the constraint bounds. For example, the OUU design point pertaining to $\beta = 6$ is six standard deviations away from the expected failure manifold. The difference between the *reliability parameter* and the *factor of safety* is that the former is driven by mathematical concepts from probability theory, whereas the latter is driven by expert opinion and industry or regulatory standards. In order to graphically interpret the results, we plot the contours of the design space in Figure 8.20, on a $25 \times 25$ Cartesian grid of the design variable bounds. The trajectory of the design points, for increasing values of the reliability parameter $\beta$, is plotted along with the mass and failure contours. The deterministic optimization case took under a minute to converge, whereas the OUU cases took between $5 - 25$ minutes to converge:

*Table 8.9: Designs resulting from the deterministic and probabilistic optimization of the flexible manipulator system.*

| Quantity | Deterministic | $\beta = 0$ | $\beta = 1$ | $\beta = 2$ | $\beta = 3$ | $\beta = 4$ | $\beta = 5$ | $\beta = 6$ | $\beta = 7$ |
|---|---|---|---|---|---|---|---|---|---|
| width 1 [m] | 0.250 | 0.250 | 0.250 | 0.271 | 0.303 | 0.343 | 0.385 | 0.428 | 0.471 |
| width 2 [m] | 0.250 | 0.250 | 0.250 | 0.250 | 0.250 | 0.278 | 0.311 | 0.347 | 0.381 |
| constraint % | 76.8 | 72.9 | 92.6 | 100 | 100 | 100 | 100 | 100 | 100 |
| $\mathbb{E}$[failure] | – | 0.729 | 0.729 | 0.650 | 0.552 | 0.482 | 0.431 | 0.387 | 0.353 |
| iterations | 7 | 7 | 7 | 9 | 8 | 34 | 62 | 43 | 41 |

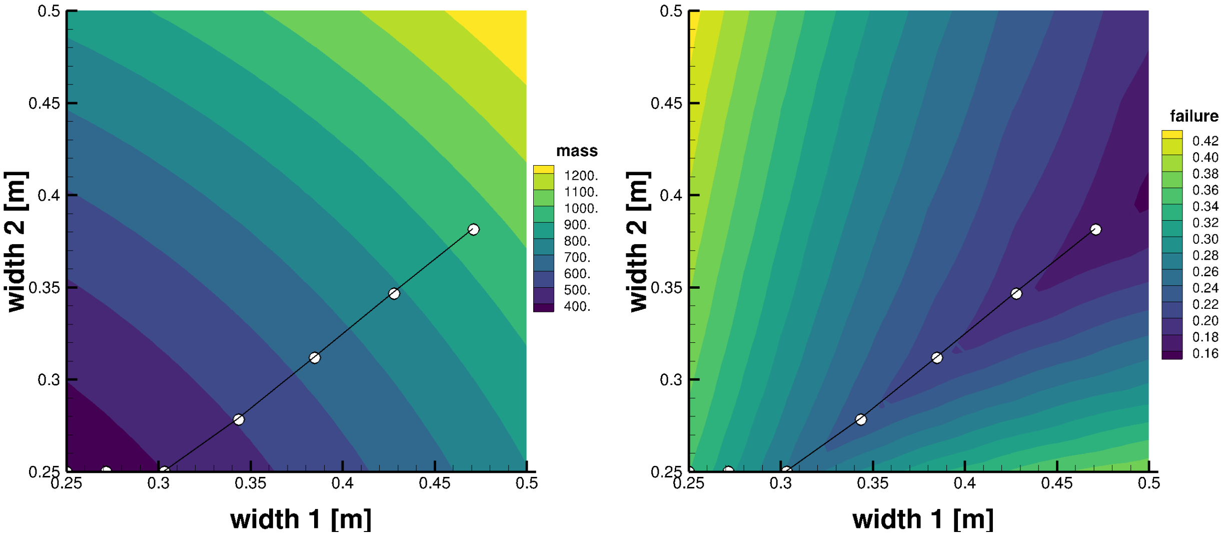


*Figure 8.20: The visualization of optimization design space with contours of the mass and failure.*

the number of optimizer iterations, function and gradient evaluations varied between these cases.

### *8.4.2.3 Scalability Studies*

The stochastic problem size grows linearly with increasing number of terms in the basis set. In this section, we perform a study that identifies the rate at which the computational effort grows with increasing problem size $N$. We use the Canadarm case, to perform forward analysis in time domain, and adjoint sensitivity analysis. We perform stochastic Galerkin projection with increasing number of terms in the expansion of the random variable $y \sim \mathcal{N}(\mu = 100,000\ kg, \sigma = 50,000\ kg, N)$, where $N = 1, \ldots, 10$. Recall, that the number of deterministic degrees of freedom in the problem is 432: therefore, the total number of stochastic degrees of freedom is $432 \times N$. The results are plotted in Figure 8.21. The slope of line is approximately 2.88, which implies that the computational effort grows as $\mathcal{O}(N^{2.88})$. The actual wall times were divided by the number of time steps taken in the simulation for normalization. In order to improve the scalability with respect to the problem size, the exploitation of symmetry and sparsity of the Jacobian, optimization of the number of quadrature nodes based on the polynomial degree of the integrand, application of sparse quadrature methods and matrix-free implementations are being considered for further studies.

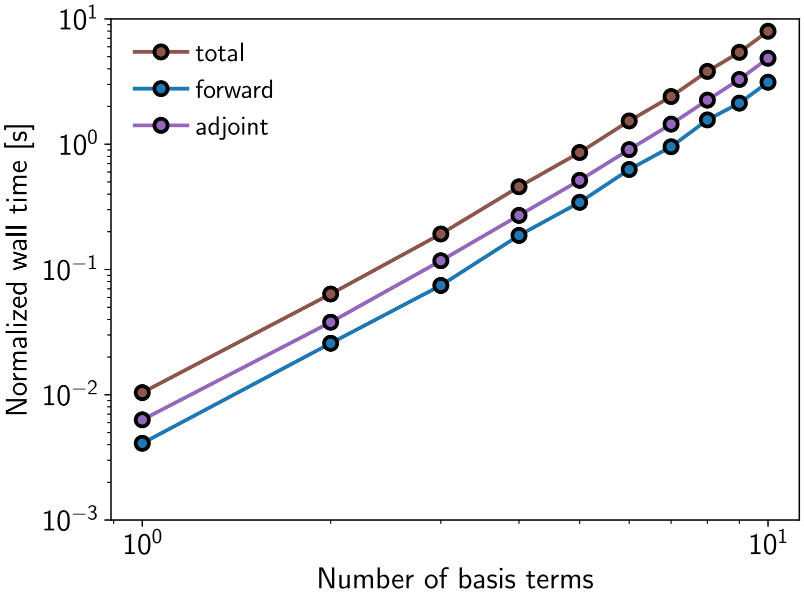


*Figure 8.21: Plot of normalized wall time versus the cardinality of the probabilistic basis set.*

**Summary.** In this chapter, we applied the semi-intrusive uncertainty propagation technique on a wide range of problems from simple ODEs to complex space robotic systems. Although, we presented the explicit details of the Galerkin projection for some test cases, in order to illustrate it from the perspective of linear algebra and inner products, all the computer implementations rely on abstractions and implicit formation of stochastic quantities from deterministic quantities as described in Section 7.2. We used the sampling method to verify the probabilistic moments, and the complex-step method to verify the adjoint derivatives. We used the four-bar mechanism and the remote manipulator model, to demonstrate the use of OUU methods to produce designs that contain probabilistic information that can be used for certification and quality assurance purposes. We demonstrated the propagation of uncertainties through time dependent physics and adjoint formulations in the context of stochastic Galerkin method.

# Part IV

# Conclusions

# CHAPTER 9
# CONTRIBUTIONS AND FUTURE WORK

*Equations Are Art inside a Mathematician's Brain.*

*Unknown Admirers*

In this Chapter, we summarize the contributions of the thesis and outline future research directions.

## 9.1 Summary of Contributions

We developed an optimization under uncertainty framework featuring

- the analysis of time dependent physics using implicit time marching methods,
- time dependent discrete adjoint based gradient evaluation, and
- the propagation of uncertainties using Galerkin projection and sampling.

We envisioned and structured the stochastic analysis capabilities as an extension of deterministic design capabilities, and therefore separated the treatment of probabilistic domain. The structuring of packages that form the developed UQ-OUU framework is shown in Figure 9.1. The time marching capabilities and adjoint method was implemented in the `TACS` finite element framework [1]. The evaluation of orthonormal basis and quadrature needed for stochastic projection and sampling are implemented as a separate package `PSPACE` [2]. The Stochastic TACS framework `STACS`[3] is an object-oriented extension of the TACS finite element framework providing implementations of `Element` and `Function` interfaces needed for stochastic finite element computations. The source code of `PSPACE` and `STACS` are written

[1] https://github.com/gjkennedy/tacs
[2] https://github.com/komahanb/pspace
[3] https://github.com/komahanb/stacs

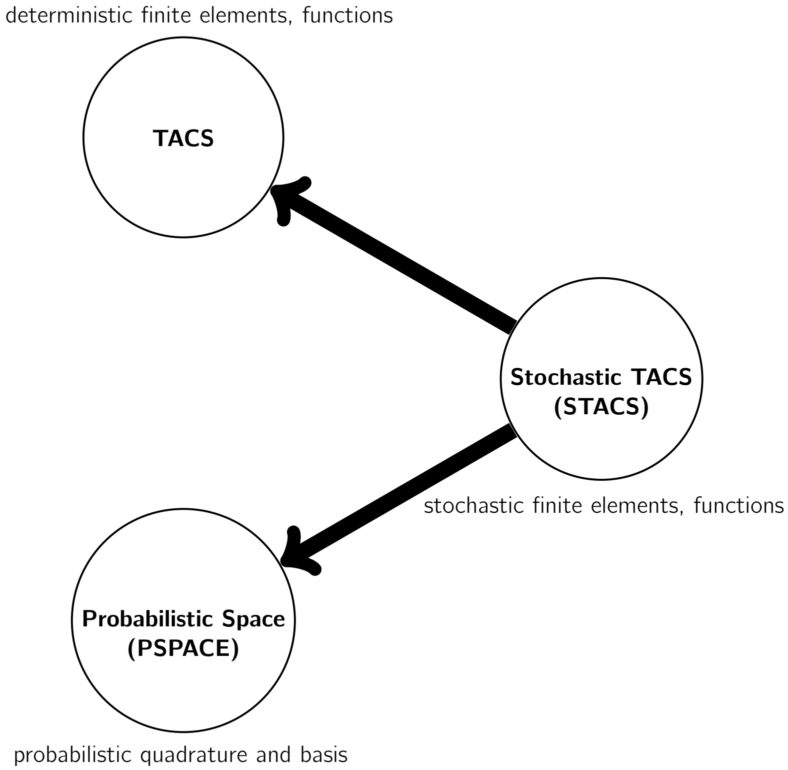

deterministic finite elements, functions
TACS
Stochastic TACS
(STACS)
stochastic finite elements, functions
Probabilistic Space
(PSPACE)
probabilistic quadrature and basis

*Figure 9.1: The open source software packages that are a part of the developed UQ-OUU framework.*

in `Fortran/C++` and wrapped in `Python`. The detailed technical contributions within this framework is outlined in the remainder of this section.

### 9.1.1 Implicit Time Marching Methods in Natural Form

The governing equations of flexible multibody dynamics are a set of second-order differential algebraic equations. We enhanced the existing implicit time marching methods such as Newmark, Runge–Kutta, Backward difference formulas, and Adams–Bashforth–Moulton method for flexible multibody dynamics, and in general for second-order differential equations. These techniques were developed based on the second-order form of equations with an abstract representation of the residuals as $R(t, \xi, u(t, \xi), \dot{u}(t, \xi), \ddot{u}(t, \xi))$, paving way for a unified implementation within the TACS framework, as well as applicability for other time dependent physics.

### 9.1.2 Time Dependent Discrete Adjoint Formulation

Using abstractions of the governing equations $R(t, \xi, u(t, \xi), \dot{u}(t, \xi), \ddot{u}(t, \xi))$ and the metrics of interest $F(t, \xi, u(t, \xi), \dot{u}(t, \xi), \ddot{u}(t, \xi))$, we derived the time dependent discrete-adjoint equations. These mathematical developments were numerically verified using the complex-step method. The abstract mathematical developments of the implicit time marching methods and their corresponding adjoint formulations, facilitated the modular and extensible programming implementation within the TACS framework.

### 9.1.3 Semi-intrusive Stochastic Galerkin Projection

Stochastic Galerkin projection techniques are used to propagate uncertainties through simulations governed by differential equations. The stochastic Galerkin methods are often challenging to implement within existing deterministic finite-element libraries as they require extensive source code modifications. In this work, we presented a semi-intrusive stochastic Galerkin methodology that enables us to reuse existing deterministic finite-element implementations to perform projection in the probabilistic domain. Furthermore, the proposed semi-intrusive method enables the use of deterministic adjoint capabilities for setting up the stochastic adjoint equations. The principal idea is to project the deterministic quantities such as residuals, Jacobians, boundary conditions, and adjoint terms on to the probabilistic space, prior to assembly of the stochastic finite element or adjoint system, while assuming the deterministic implementations to be black-box. In order for the proposed method to work, the deterministic implementations must be able to recompute deterministic quantities for different values of probabilistically modeled parameters. The proposed semi-intrusive stochastic Galerkin approach is demonstrated within the assembly and solution architecture of TACS – a finite-element framework with adjoint-based gradient evaluation methods, with problems from flexible multibody dynamics.

### 9.1.4 Flexible Multibody Dynamics Applications

The mathematical techniques and algorithms developed in this work are demonstrated using a wide range of problems from simple ODE models to complex flexible multibody systems. The rotorcraft hub dynamics was analyzed by modeling the full control chain containing translational and rotational actuators, swash plates and blades. The adjoint gradient based optimization demonstration was carried out using this model. We demonstrated the semi-intrusive UQ-OUU capabilities on the mechanism modeled after the *Canadarm*. It was shown the in the presence of uncertainties in payloads, the system experiences stresses that have a large variability. We showed that UQ can be a great tool in assessing the risk associated with such operating conditions.

## 9.2 Future Work

The suggestions for future work are outlined as follows. Some of these suggestions are exciting theoretical/mathematical endeavors while others are applications of the developed capabilities for OUU of aeromechanical systems.

### 9.2.1 Mathematical Formalisms of Implicit Time Marching and Sensitivity Analysis

- In areas of mathematical physics such as chaotic dynamics, the governing equations contain as high as sixth-order derivatives in time (see Chlouverakis and Sprott [20]). In this context, the development of time marching methods based on abstract natural forms is of interest. In this case, we would be considering a system of $d$-th order nonlinear ordinary differential equations in time $t$ written in abstract implicit form

$$R(q^{(d)}(t), \ldots, q^{(1)}(t), q^{(0)}(t), t) = 0, \tag{9.1}$$

  where $q^{(j)}(t) := \frac{d^j q(t)}{dt^j}$ $\forall j = 0, \ldots, d$ are the field variables and their time derivatives. The implicit solution techniques in natural form can be developed by "generalizing" the Newton–Raphson iterative procedure described in Chapter 4.

- The development of time dependent direct sensitivity analysis equations in the context of Newmark, BDF, ABM and Runge–Kutta, and perhaps other methods is an interesting endeavor to understand the full spectrum of semianalytical methods for sensitivity analysis along with the adjoint counterparts. An open-source implementation of implicit time marching methods and corresponding semianalytical sensitivity analysis formulations, can be a useful tool for the scientific community.

- The extension of the time dependent adjoint and direct sensitivity analysis equations to higher-order equations (9.1), can provide great intuitions on the mathematical structure. In this thesis, graphical illustrations of the adjoint equations in Chapter 4 were presented to elucidate the intricate structure of adjoint equations.

### 9.2.2 Semi-intrusive Uncertainty Propagation for Finite Volume Frameworks

The use of mathematical abstraction for derivations, often, allows great flexibility in using them in areas formerly unintended. The abstract development of stochastic Galerkin projection equations facilitates the direct application of the semi-intrusive technique to problems using the finite volume method (FVM) for the treatment of spatial derivatives. Recall that *cells* are the fundamental units of FVM computations, and *elements* are the fundamental units of FEM framework. As outlined in Section 7.3, the methods can be extended to cell-wise computations performed with FVM. In this case, we ought to proceed with the following interpretation of residual as cell-wise residuals.

$$\begin{gathered}\left\langle \widehat{\psi}_i^y(y) \;\middle|\; R^c\left(t, y, u^c(t,y), \dot{u}^c(t,y)\,, \ddot{u}^c(t,y)\right)_{\rho^y(y)}^{\mathcal{Y}} \right\rangle \\ \approx \\ \sum_{q=1}^{Q} \alpha_q \widehat{\psi}_i^y(y_q) \underbrace{R^c\left(t, y_q, u^c(t,y_q), \dot{u}^c(t,y_q), \ddot{u}^c(t,y_q)\right)}_{\text{cell-wise deterministic residuals for } y_q}\end{gathered} \tag{9.2}$$

Again, one has the alternative of projecting system-wide assembled residuals, if it is simpler to implement.

### 9.2.3 Algebraic Multigrid for Stochastic Galerkin Computations

The stochastic Galerkin Jacobian matrices are shown to have interesting sparsity patterns arising from the nonlinearity of problem parameters dependent on random variables. The investigation of algebraic multigrid (AMG) techniques for partial differential equations has received good attention [129]. This naturally guides us to investigate AMG techniques to accelerate the solution process to bigger linear systems in the context of stochastic PDEs. The implementation of AMG can be simplified, due the simpler implicit construction of stochastic Jacobians from deterministic Jacobians.

### 9.2.4 Topology Optimization Under Uncertainty

The application of OUU methods in the context of topology optimization is receiving attention among researchers [130–140]. Maute [140] mentions the importance of considering probabilistic variations in material properties, geometry, boundary conditions to produce robust and reliable designs. Guest and Igusa [139] considers the uncertainties in loading conditions for topology optimization. The expectation and variance of compliance are used as the objective by Dunning and Kim [136]. The works in the literature use nonintrusive sampling based methods or simpler assessments for quantifying the effect of uncertainties. The lack of published works on stochastic Galerkin projection based topology optimization, serves to affirm the difficulty in the development of such frameworks, that is, the intrusiveness is a big hurdle for the adaptation of SGM as preferred method for UQ applications. The semi-intrusive technique for stochastic Galerkin projection aligns well with addressing this difficulty and can serve as an easier to implement method on top of deterministic frameworks for topology optimization.

### 9.2.5 Multidisciplinary Optimization Under Uncertainty

The incorporation of UQ techniques within multidisciplinary analysis and optimization (MDAO) has been noted as one of the key interests in the *NASA's Vision for 2030* [141]. The application of the semi-intrusive projection technique to existing deterministic tools, can enable an easier integration of UQ into MDAO frameworks. We recall that the requirement for the semi-intrusive technique is that the deterministic tools ought to have the flexibility to update problem parameters that are modeled as random.

# BIBLIOGRAPHY

[1] O. A. Bauchau, P. Betsch, A. Cardona, J. Gerstmayr, B. Jonker, P. Masarati, and V. Sonneville. Validation of flexible multibody dynamics beam formulations using benchmark problems. *Multibody System Dynamics*, 37(1):29–48, may 2016. ISSN 1573-272X. doi:10.1007/s11044-016-9514-y. URL `https://doi.org/10.1007/s11044-016-9514-y`.

[2] A. Keane and P. Nair. *Computational Approaches for Aerospace Design*. John Wiley & Sons, 2005.

[3] G. J. Kennedy and J. R. R. A. Martins. A parallel finite-element framework for large-scale gradient-based design optimization of high-performance structures. *Finite Elements in Analysis and Design*, 87(0):56 – 73, 2014. ISSN 0168-874X. doi:10.1016/j.finel.2014.04.011.

[4] O. A. Bauchau. *Flexible Multibody Dynamics*. Springer Netherlands, 2011. ISBN 978-94-007-0334-6. doi:10.1007/978-94-007-0335-3.

[5] L. Meirovitch. *Principles and techniques of vibrations*. Prentice Hall, Upper Saddle River, NJ, 1st edition, 1997. ISBN 0023801417.

[6] E. J. Haug, D. Negrut, and M. Iancu. *Implicit Integration of the Equations of Multibody Dynamics*, pages 242–267. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. ISBN 978-3-662-03729-4. doi:10.1007/978-3-662-03729-4_11. URL `http://dx.doi.org/10.1007/978-3-662-03729-4_11`.

[7] O. A. Bauchau and A. Laulusa. Review of Contemporary Approaches for Constraint Enforcement in Multibody Systems. *ASME. J. Appl. Mech.*, 3, No. 1:1–8, 2007. doi:10.1115/1.2803258.

[8] G. D. Byrne and A. C. Hindmarsh. A polyalgorithm for the numerical solution of ordinary differential equations. *ACM Transactions on Mathematical Software (TOMS)*, 1(1):71–96, 1975. ISSN 0098-3500. doi:10.1145/355626.355636.

[9] A. C. Hindmarsh. {ODEPACK}, A Systematized Collection of {ODE} Solvers. *IMACS Transactions on Scientific Computation*, 1:55–64, 1983.

[10] K. Brenan, S. Campbell, and L. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, 1995. doi:10.1137/1.9781611971224.

[11] L. Fox and E. T. Goodwin. Some new methods for the numerical integration of ordinary differential equations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 45(3):373–388, 007 1949. doi:10.1017/S0305004100025007.

[12] N. M. Newmark. A method of computation for structural dynamics. *Journal of the Eng. Mech. Div.*, 85(3), 1959.

[13] J. Chung and G. M. Hilbert. A Time Integration Algorithm for Structural Dynamics With Improved Numerical Dissipation: The Generalized-$\alpha$ Method. *ASME. J. Appl. Mech.*, 60, No. 2:371–375, 1993. doi:10.1115/1.2900803.

[14] C. F. Curtiss and J. O. Hirschfelder. Integration of Stiff Equations. *Proceedings of the National Academy of Sciences of the United States of America*, 38(3):235–243, 1952.

[15] P. Henrici. *Discrete variable methods in ordinary differential equations.* Wiley, New York, 1962.

[16] A. Jameson, W. Schmidt, and E. Turkel. *Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes.* doi:10.2514/6.1981-1259. URL `https://arc.aiaa.org/doi/abs/10.2514/6.1981-1259`.

[17] 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, dec 2015. ISSN 0001-1452. doi:10.2514/1.J053813. URL `https://doi.org/10.2514/1.J053813`.

[18] M. Cerquaglia, D. Thomas, R. Boman, V. Terrapon, and J.-P. Ponthot. A fully partitioned Lagrangian framework for FSI problems characterized by free surfaces, large solid deformations and displacements, and strong added-mass effects. *Computer Methods in Applied Mechanics and Engineering*, 348:409 – 442, 2019. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2019.01.021. URL `http://www.sciencedirect.com/science/article/pii/S0045782519300428`.

[19] O. Bauchau, C. Bottasso, and Y. Nikishkov. Modeling rotorcraft dynamics with finite element multibody procedures. *Mathematical and Computer Modelling*, 33(10):1113 – 1137, 2001. ISSN 0895-7177. doi:10.1016/S0895-7177(00)00303-4.

[20] K. E. Chlouverakis and J. C. Sprott. Chaotic hyperjerk systems. *Chaos, Solitons & Fractals*, 28(3):739–746, 2006. ISSN 0960-0779. doi:https://doi.org/10.1016/j.chaos.2005.08.019. URL `http://www.sciencedirect.com/science/article/pii/S0960077905006776`.

[21] F. Bashforth and J. C. Adams. An attempt to test the theories of capillary action by comparing the theoretical and measured forms of drops of fluid, with an explanation of the method of integration employed in constructing the tables which give the theoretical forms of such drops. Cambridge University Press, 1883.

[22] F. R. Moulton. *New Methods in Exterior Balistics.* University of Chicago Press, 1926.

[23] J. C. Butcher. Implicit Runge–Kutta Processes. *Mathematics of Computation*, 18(85): 50–64, 1964. ISSN 00255718, 10886842. doi:10.1090/S0025-5718-1964-0159424-9.

[24] R. Alexander. Diagonally Implicit Runge–Kutta Methods for Stiff O.D.E.'s. *SIAM J. Numer. Anal.*, 14(6):1006–1021, 1977. doi:10.1137/0714068.

[25] J. Cash. Diagonally implicit Runge–Kutta formulae for the numerical integration of nonlinear two-point boundary value problems. *Computers & Mathematics with Applications*, 10(2):123 – 137, 1984. ISSN 0898-1221. doi:10.1016/0898-1221(84)90043-9.

[26] C. W. Gear. Simultaneous Numerical Solution of Differential-Algebraic Equations. *IEEE Transactions on Circuit Theory*, 18(1):89–95, Jan 1971. ISSN 0018-9324. doi:10.1109/TCT.1971.1083221.

[27] W. Squire and G. Trapp. Using complex variables to estimate derivatives of real functions. *SIAM Review*, 40(1):110–112, 1998. doi:10.1137/S003614459631241X.

[28] J. R. R. A. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software*, 29(3):245–262, Sept. 2003. doi:10.1145/838250.838251.

[29] J. Fike and J. Alonso. *The Development of Hyper-Dual Numbers for Exact Second-Derivative Calculations.* doi:10.2514/6.2011-886. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2011-886`.

[30] J. Fike, S. Jongsma, J. Alonso, and E. V. D. Weide. *Optimization with Gradient and Hessian Information Calculated Using Hyper-Dual Numbers.* doi:10.2514/6.2011-3807. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2011-3807`.

[31] L. L. Sherman, A. C. Taylor III, L. L. Green, and P. A. Newman. First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods. *Journal of Computational Physics*, 129:307 – 331, 1996. doi:10.2514/6.1994-4262.

[32] L. Hascoët. TAPENADE: a tool for Automatic Differentiation of programs. In *Proceedings of 4th European Congress on Computational Methods, ECCOMAS'2004, Jyvaskyla, Finland*, 2004.

[33] D. P. Ghate and M. B. Giles. Efficient Hessian Calculation using Automatic Differentiation. AIAA Paper, 2007-4059, June, 2007.

[34] M. P. Rumpfkeil and D. J. Mavriplis. Efficient Hessian Calculations using Automatic Differentiation and the Adjoint Method with Applications. *AIAA Journal*, 48, No. 10: 2406–2417, 2010. doi:10.2514/1.J050451.

[35] G. Hou, S. Arunkumar, and N. S. Tiwari. First- and second-order sensitivity analysis of finite element equations via automatic differentiation. AIAA Paper, 98-4764, August, 1998.

[36] L. C. Dixon. Automatic Differentiation: Calculation of the Hessian. In *Encyclopedia of Optimization*, pages 82–86. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[37] C. H. Bischof, H. M. Bücker, P. Hovland, U. Naumann, and J. Utke. *Advances in Automatic Differentiation*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 3540689354, 9783540689355. doi:10.1007/978-3-540-68942-3.

[38] L. L. Green, P. A. Newman, and K. J. Haigler. Sensitivity Derivatives for Advanced CFD Algorithm and Viscous Modelling Parameters via Automatic Differentiation. *Journal of Computational Physics*, 125(2):313–324, 1996. doi:10.2514/6.1993-3321.

[39] I. M. Jin and L. A. Schmit. Control design variable linking for optimization of structural/control systems. *AIAA Journal*, 30, No. 7:1892–1900, 1992. doi:10.2514/3.11152.

[40] A. D. Belegundu and J. S. Arora. A sensitivity interpretation of adjoint variables in optimal design. *Computer Methods in Applied Mechanics and Engineering*, 48(1): 81–89, 1985. ISSN 00457825. doi:10.1016/0045-7825(85)90068-4.

[41] H. M. Adelman and R. T. Haftka. Sensitivity Analysis of Discrete Structural Systems. *AIAA Journal*, 24(5):823–832, 1986. doi:10.2514/3.48671.

[42] R. Haftka and H. Adelman. Recent developments in structural sensitivity analysis. *Structural optimization*, 1(3):137–151, 1989. ISSN 0934-4373. doi:10.1007/BF01637334.

[43] K. D. Bhalerao, M. Poursina, and K. S. Anderson. An efficient direct differentiation approach for sensitivity analysis of flexible multibody systems. *Multibody System Dynamics*, 23(2):121–140, 2009. ISSN 1573-272X. doi:10.1007/s11044-009-9176-0.

[44] D. Dopico, Y. Zhu, A. Sandu, and C. Sandu. Direct and Adjoint Sensitivity Analysis of Ordinary Differential Equation Multibody Formulations. *Journal of Computational and Nonlinear Dynamics*, 10(1):1–7, 2014. doi:10.1115/1.4026492.

[45] G. Kreisselmeier and R. Steinhauser. Systematic control design by optimizing a vector performance index. In *International Federation of Active Controls Symposium on Computer-Aided Design of Control Systems*, Zurich, Switzerland, 1979.

[46] G. J. Kennedy and J. E. Hicken. Improved constraint-aggregation methods. *Computer Methods in Applied Mechanics and Engineering*, 289:332 – 354, 2015. ISSN 0045-7825. doi:10.1016/j.cma.2015.02.017.

[47] M. A. Akgun, R. T. Haftka, K. C. Wu, J. L. Walsh, and J. H. Garcelon. Efficient Structural Optimization for Multiple Load Cases Using Adjoint Sensitivities. *AIAA Journal*, 39(3):511–516, 2001. doi:10.2514/2.1336.

[48] G. W. Burgreen and O. Baysal. Three-Dimensional Aerodynamic Shape Optimization Using Discrete Sensitivity Analysis. *AIAA Journal*, 34, No. 9:1761–1770, 1996. doi:10.2514/3.13305.

[49] W. Anderson and V. Venkatakrishnan. Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation. *Computers & Fluids*, 28(4âĂŞ5):443 – 480, 1999. ISSN 0045-7930. doi:http://dx.doi.org/10.1016/S0045-7930(98)00041-3.

[50] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3(3):233–260, 1988. ISSN 08857474. doi:10.1007/BF01061285.

[51] D. I. Papadimitriou and K. C. Giannakoglou. Computation of the Hessian matrix in aerodynamic inverse design using continuous adjoint formulations. *Computers and Fluids*, 37:1029 – 1039, 2008.

[52] B. Grossman, Z. Gürdal, R. T. Haftka, G. J. Strauch, and W. M. Eppard. Integrated aerodynamic/structural design of a sailplane wing. *Journal of Aircraft*, 25(9):855–860, 2013/09/28 1988. doi:10.2514/3.45980.

[53] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. A coupled–adjoint sensitivity analysis method for high–fidelity aero–structural design. *Optimization and Engineering*, 6: 33–62, 2005. doi:10.1023/B:OPTE.0000048536.47956.62.

[54] Y. Cao, S. Li, L. Petzold, and R. Serban. Adjoint Sensitivity Analysis for Differential-Algebraic Equations: The Adjoint DAE System and Its Numerical Solution. *SIAM Journal on Scientific Computing*, 24(3):1076–1089, 2003. doi:10.1137/S1064827501380630.

[55] J.-Y. Ding, Z.-K. Pan, and L.-Q. Chen. Second order adjoint sensitivity analysis of multibody systems described by differential–algebraic equations. *Multibody System Dynamics*, 18(4):599–617, 2007. ISSN 1573-272X. doi:10.1007/s11044-007-9080-4.

[56] K. Nachbagauer, S. Oberpeilsteiner, K. Sherif, and W. Steiner. The Use of the Adjoint Method for Solving Typical Optimization Problems in Multibody Dynamics. *Journal of Computational and Nonlinear Dynamics*, 10(6):1–10, 2015. ISSN 1555-1415. doi:10.1115/1.4028417.

[57] R. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A Spectral Approach (2nd edition)*. New York: Springer, 1991.

[58] H. G. Matthies. *Uncertainty Quantification with Stochastic Finite Elements*, chapter 27. John Wiley & Sons, 2007. ISBN 9780470091357. doi:10.1002/0470091355.ecm071. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/0470091355.ecm071`.

[59] M. A. GutiÃl'rrez and S. Krenk. *Stochastic Finite Element Methods*, chapter 20. John Wiley & Sons, 2004. ISBN 9780470091357. doi:10.1002/0470091355.ecm044. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/0470091355.ecm044`.

[60] D. Xiu. *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press, 2010. ISBN 978-0691142128.

[61] O. P. L. Maître and O. M. Knio. *Spectral Methods for Uncertainty Quantification, Scientific Computation*. Springer, 2010. ISBN 978-9048135196.

[62] R. Ghanem, H. Owhadi, and D. Higdon. *Handbook of uncertainty quantification*. 06 2017. doi:10.1007/978-3-319-12385-1.

[63] C. Soize. *Uncertainty Quantification*, volume 47. 01 2017. doi:10.1007/978-3-319-54339-0.

[64] M. Gunzburger. *An Applied/Computational MathematicianâĂŹs View of Uncertainty Quantification for Complex Systems: Proceedings of 2015 and 2016 ACMES Conferences*, pages 133–151. 01 2019. ISBN 978-1-4939-9050-4. doi:10.1007/978-1-4939-9051-1_5.

[65] S. Hosder, R. W. Walters, and R. Perez. A non-intrusive polynomial chaos method for uncertainty propagation in CFD simulations. AIAA Paper, 2006-891, 2006.

[66] M. S. Elred, C. G. Webster, and P. G. Constantine. Evaluation of Non-Intrusive Approaches for Wiener-Askey Generalized Polynomial Chaos. AIAA Paper, 2008-1892, 2008.

[67] B. A. Jones, A. Doostan, and G. H. Born. Nonlinear Propagation of Orbit Uncertainty Using Non-Intrusive Polynomial Chaos. *AIAA Journal of Guidance, Control, and Dynamics*, 36, No.2:415–425, 2013.

[68] G. Fishman. *Monte-Carlo: Concepts, Algorithms, and Applications*. New York: Springer-Verlag, 1996.

[69] B. Peherstorfer, M. Gunzburger, and K. Willcox. Convergence analysis of multifidelity Monte Carlo estimation. *Numerische Mathematik*, 139:1–25, 01 2018. doi:10.1007/s00211-018-0945-7.

[70] M. S. Elred. Recent Advances in Non-Intrusive Polynomial Chaos and Stochastic Collocation Methods for Uncertainty Analysis and Design. AIAA Paper, 2009-2274, 2009.

[71] H. Cheng and A. Sandu. Collocation least-squares polynomial chaos method. SCS/ACM, 2010.

[72] D. Xiu. Fast Numerical Methods for Stochastic Computations: A Review. *Comm. Comput. Phys.*, 5, No. 2–4:242–272, 2009.

[73] M. Gunzburger, C. G. Webster, and G. Zhang. *Sparse Collocation Methods for Stochastic Interpolation and Quadrature*, pages 717–762. 06 2017. doi:10.1007/978-3-319-12385-1_29.

[74] O. Roderick, M. Anitescu, and P. Fischer. Polynomial Regression Approaches Using Derivative Information for Uncertainty Quantification. *J. of Nuclear Science and Engineering*, 164, No.2:122–139, 2010.

[75] Y. Li, M. Anitescu, O. Roderick, and F. Hickernell. Orthogonal Bases for Polynomial Regression with Derivative Information in Uncertainty Quantification. *International Journal for Uncertainty Quantification*, 1, No.4:297–320, 2011.

[76] M. P. Rumpfkeil. Optimizations Under Uncertainty Using Gradients, Hessians, and Surrogate Models. *AIAA Journal*, 51, No. 2:444–451, 2013.

[77] M. P. Rumpfkeil. Robust design under mixed aleatory/epistemic uncertainties using gradients and surrogates. *Journal of Uncertainty Analysis and Applications*, 1(1):7, Oct 2013. ISSN 2195-5468. doi:10.1186/2195-5468-1-7. URL `https://doi.org/10.1186/2195-5468-1-7`.

[78] K. Boopathy and M. P. Rumpfkeil. *Robust Optimizations of Structural and Aerodynamic Designs*. doi:10.2514/6.2014-2595. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2014-2595`.

[79] K. Boopathy, M. P. Rumpfkeil, and R. M. Kolonay. *Robust Optimization of a Wing Under Structural and Material Uncertainties*. doi:10.2514/6.2015-0920. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2015-0920`.

[80] T. Chatterjee, R. Chowdhury, and P. Ramu. Decoupling uncertainty quantification from robust design optimization. *Structural and Multidisciplinary Optimization*, 59(6): 1969–1990, Jun 2019. ISSN 1615-1488. doi:10.1007/s00158-018-2167-0. URL `https://doi.org/10.1007/s00158-018-2167-0`.

[81] O. Ernst and E. Ullmann. Stochastic Galerkin Matrices. *SIAM Journal on Matrix Analysis and Applications*, 31(4):1848–1872, 2010. doi:10.1137/080742282. URL `https://doi.org/10.1137/080742282`.

[82] P. Delgado and V. Kumar. A stochastic Galerkin approach to uncertainty quantification in poroelastic media. *Applied Mathematics and Computation*, 266:328–338, 2015. ISSN 0096-3003. doi:https://doi.org/10.1016/j.amc.2015.04.127. URL `http://www.sciencedirect.com/science/article/pii/S0096300315006001`.

[83] D. Xiu and G. E. Karniadakis. The Wiener-Askey Polynomial Chaos for Stochastic Differential Equations. *SIAM Journal of Scientific Computing*, 24(2):619–644, 2002.

[84] R. Ghanem and J. Red-Horse. Propagation of Probabilistic Uncertainty in Complex Physical Systems Using a Stochastic Finite Element Approach. *Physica D*, 133:137–144, 1999.

[85] R. Ghanem. Ingredients for a general purpose stochastic finite elements implementation. *Computer Methods in Applied Mechanics and Engineering*, 168(1):19 – 34, 1999. ISSN 0045-7825. doi:https://doi.org/10.1016/S0045-7825(98)00106-6. URL `http://www.sciencedirect.com/science/article/pii/S0045782598001066`.

[86] M. D. Gunzburger, C. G. Webster, and G. Zhang. Stochastic finite element methods for partial differential equations with random input data. *Acta Numerica*, 23:521–650, 05 2014. doi:10.1017/S0962492914000075.

[87] R. Abgrall and S. Tokareva. The Stochastic Finite Volume Method. *SEMA SIMAI Springer Series*, pages 1–57, 01 2017. doi:10.1007/978-3-319-67110-9_1.

[88] G. Taguchi. *Quality Engineering through Design Optimization.* Kraus International Publications, New York, 1984.

[89] G. Taguchi. *Introduction to Quality Engineering.* American Supplier Institute, 1989.

[90] P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications.* Kluwer, 1997.

[91] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton Series in Applied Mathematics, Princeton University Press, 2009.

[92] S. Sundaresan, K. Ishii, and D. R. Houser. A ROBUST OPTIMIZATION PROCEDURE WITH VARIATIONS ON DESIGN VARIABLES AND CONSTRAINTS. *Engineering Optimization*, 24(2):101–117, 1995. doi:10.1080/03052159508941185. URL `https://doi.org/10.1080/03052159508941185`.

[93] M. M. Putko, A. C. Taylor III, P. A. Newmann, and L. L. Green. Approach for Input Uncertainty Propagation and Robust Design in CFD Using Sensitivity Derivatives. *Journal of Fluids Engineering*, 124(1):60–69, 2002.

[94] C. R. Gumbert, P. A. Newman, and G. J. Hou. Effect of random geometric uncertainty on the computational design of 3-D wing. AIAA Paper, 2002-2806, June, 2002.

[95] Y. Cao, M. Y. Hussaini, and T. A. Zang. An efficient Monte Carlo method for optimal control problems with uncertainty. *Computational Optimization and Applications*, 26(3):219–230, 2003.

[96] J. M. Luckring, M. J. Hemsch, and J. H. Morrison. Uncertainty in computational aerodynamics. AIAA Paper, 2003-0409, January, 2003.

[97] Y. Cao, M. Y. Hussaini, T. A. Zang, and A. Zatezalo. A variance reduction method based on sensitivity derivatives. *Applied Numerical Mathematics*, 56:800–813, 2006.

[98] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. Foreword: special issue on robust optimization. *Mathematical Programming*, 107(1-2):1–3, 2006.

[99] F. Chalot, Q. Dinh, E. Herbin, L. Martin, M. Ravachol, and G. Roge. Estimation of the impact of geometrical uncertainties on aerodynamic coefficients using CFD. AIAA Paper, 2068-2008, April, 2008.

[100] W. Chen, J. Allen, K. Tsui, and F. Mistree. Procedure for Robust Design: Minimizing Variations Caused by Noise Factors and Control Factors. *Journal of Mechanical Design*, 118(4):478–485, 1996.

[101] W. Chen and X. Du. Towards a Better Understanding of Modeling Feasibility Robustness in Engineering Design. *Journal of Mechanical Design*, 122(4):385–394, 1999.

[102] K. Zaman, M. McDonald, S. Mahadevan, and L. Green. Robustness-based Design Optimization Under Data Uncertainty. *Structural and Multidisciplinary Optimization*, pages 1–15, 2011.

[103] Z. Mourelatos and J. Liang. A Methodology for Trading-off Performance and Robustness Under Uncertainty. *Journal of Mechanical Design*, 128:856, 2006.

[104] G. Park, T. Lee, H. Kwon, and K. Hwang. Robust Design: an Overview. *AIAA Journal*, 44(1):181–191, 2006.

[105] X. Du and W. Chen. Methodology for Managing the Effect of Uncertainty in Simulation-Based Design. *AIAA Journal*, 38(8):1471–1478, 2000.

[106] D. Padmanabhan, H. Agarwal, J. Renaud, and S. Batill. A study using Monte Carlo Simulation for failure probability calculation in Reliability-Based Optimization. *Optimization and Engineering*, 7(3):297–316, 2006. ISSN 1389-4420. doi:10.1007/s11081-006-9973-8.

[107] R. Paiva, A. Carvalho, C. Crawford, A. Suleman, and R. Canfield. A Robust and Reliability-Based Design Optimization Framework for Wing Design. AIAA Paper, 2010-2919, 2010.

[108] H. Agarwal. *Reliability Based Design Optimization: Formulations and Methodologies*. PhD thesis, University of Notre Dame, 2004.

[109] A. Parkinson, C. Sorensen, and N. Pourhassan. A general approach for robust optimal design. *Trans. ASME*, 115:74–80, 1993.

[110] X. Du and W. Chen. Towards a better understanding of mod-eling feasibility robustness in engineering design. *ASME Journal of Mechanical Design*, 122(4):385–394, 2000.

[111] S. Rangavajhala, A. Mullur, and A. Messac. The challenge of equality constraints in robust design optimization: examination and new approach. *Structural and Multidisciplinary Optimization*, 34(5):381–401, Nov 2007. ISSN 1615-1488. doi:10.1007/s00158-007-0104-8. URL `https://doi.org/10.1007/s00158-007-0104-8`.

[112] C. Lanczos. *The Variational Principles of Mechanics*. Dover, New York, 4th edition, 2015. ISBN 978-0-486-65067-8.

[113] E. N. Dvorkin and K.-J. Bathe. A continuum mechanics based four-node shell element for general nonlinear analysis. *Engineering Computations*, 1:77–88, 1984.

[114] M. L. Bucalem and K.-J. Bathe. Higher-order MITC general shell elements. *International Journal for Numerical Methods in Engineering*, 36:3729–3754, 1993. ISSN 1097-0207. doi:10.1002/nme.1620362109.

[115] G. J. Kennedy and K. Boopathy. A Scalable Adjoint Method for Coupled Flexible Multibody Dynamics. In *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA SciTech*, 2016. doi:10.2514/6.2016-1907.

[116] G. J. Kennedy. Strategies for adaptive optimization with aggregation constraints using interior-point methods. *Computers & Structures*, 153:217 – 229, 2015. ISSN 0045-7949. doi:10.1016/j.compstruc.2015.02.024.

[117] C. Geuzaine and J. F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009. ISSN 00295981. doi:10.1002/nme.2579.

[118] R. E. Perez, P. W. Jansen, and J. R. R. A. Martins. pyOpt: a Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization. *Structural and Multidisciplinary Optimization*, 45(1):101–118, 2012. doi:10.1007/s00158-011-0666-3.

[119] K. Boopathy and G. J. Kennedy. Parallel Finite Element Framework for Rotorcraft Multibody Dynamics and Discrete Adjoint Sensitivities. *AIAA Journal*, 0(0):1–14, 2019. doi:10.2514/1.J056585. URL `https://doi.org/10.2514/1.J056585`.

[120] K. Boopathy and G. Kennedy. Adjoint-based derivative evaluation methods for flexible multibody systems with rotorcraft applications. In *55th AIAA Aerospace Sciences Meeting*, Grapevine, TX, January 2017. doi:10.2514/6.2017-1671.

[121] M. Chatzimanolakis, K.-D. Kantarakias, V. Asouti, and K. Giannakoglou. A painless intrusive polynomial chaos method with RANS-based applications. *Computer Methods in Applied Mechanics and Engineering*, 348:207 – 221, 2019. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2019.01.018. URL `http://www.sciencedirect.com/science/article/pii/S0045782519300398`.

[122] C. Damaren and I. Sharf. Simulation of Flexible-Link Manipulators With Inertial and Geometric Nonlinearities. *Journal of Dynamic Systems, Measurement, and Control*, 117(1):74–87, 03 1995. ISSN 0022-0434. doi:10.1115/1.2798525. URL `https://doi.org/10.1115/1.2798525`.

[123] C. J. Damaren. Passivity analysis for flexible multilink space manipulators. *Journal of Guidance, Control, and Dynamics*, 18(2):272–279, 1995. doi:10.2514/3.21380. URL `https://doi.org/10.2514/3.21380`.

[124] C. L. Damaren. Approximate inverse dynamics and passive feedback for flexible manipulators with large payloads. *IEEE Transactions on Robotics and Automation*, 12 (1):131–138, 1996.

[125] K. B. Michael Hiltz, Craig Rice and R. Allison. CANADARM: 20 YEARS OF MISSION SUCCESS THROUGH ADAPTATION. 6th International Symposium on Artificial Intelligence and Robotics & Automation in Space, i-SAIRAS, June 18-22, 2001.

[126] J. V. Llop, J. Drew, R. Zappulla, and M. Romano. *Autonomous Capture of a Resident Space Object by a Spacecraft with a Robotic Manipulator: Analysis, Simulation and Experiments*. doi:10.2514/6.2016-5269. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2016-5269`.

[127] A. Walsh and J. R. Forbes. Modeling and Control of Flexible Telescoping Manipulators. *IEEE Transactions on Robotics*, 31(4):936–947, 2015.

[128] L. Puig, A. Barton, and N. Rando. A review on large deployable structures for astrophysics missions. *Acta Astronautica*, 67:12–26, 07 2010. doi:10.1016/j.actaastro.2010.02.021.

[129] K. StÃijben. A review of algebraic multigrid. *Journal of Computational and Applied Mathematics*, 128(1):281 – 309, 2001. ISSN 0377-0427. doi:https://doi.org/10.1016/S0377-0427(00)00516-1. URL http://www.sciencedirect.com/science/article/pii/S0377042700005161. Numerical Analysis 2000. Vol. VII: Partial Differential Equations.

[130] A. Vishwanathan, G. A. Vio, and T. Kipouros. *Considering Boundary Condition Uncertainty for Robust Topology Optimization of Flat Plates*. 2020. doi:10.2514/6.2020-1147. URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-1147.

[131] J. Richardson, R. F. Coelho, and S. Adriaenssens. A unified stochastic framework for robust topology optimization of continuum and truss-like structures. *Engineering Optimi*, 48(2):334–350, 2016. doi:10.1080/0305215X.2015.1011152. URL https://doi.org/10.1080/0305215X.2015.1011152.

[132] D. Wang and W. Gao. Robust topology optimization under load position uncertainty. *International Journal for Numerical Methods in Engineering*, 120(11):1249–1272, 2019. doi:10.1002/nme.6180. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6180.

[133] S. De, J. Hampton, K. Maute, and A. Doostan. Topology Optimization under Uncertainty using a Stochastic Gradient-based Approach, 2019.

[134] V. Keshavarzzadeh, F. Fernandez, and D. A. Tortorelli. Topology optimization under uncertainty via non-intrusive polynomial chaos expansion. *Computer Methods in Applied Mechanics and Engineering*, 318:120 – 147, 2017. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2017.01.019. URL http://www.sciencedirect.com/science/article/pii/S0045782516313019.

[135] W. Zhang and Z. Kang. Robust shape and topology optimization considering geometric uncertainties with stochastic level set perturbation. *International Journal for Numerical Methods in Engineering*, 110(1):31–56, 2017. doi:10.1002/nme.5344. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.5344.

[136] P. D. Dunning and H. A. Kim. Robust Topology Optimization: Minimization of Expected and Variance of Compliance. *AIAA Journal*, 51(11):2656–2664, 2013. doi:10.2514/1.J052183. URL https://doi.org/10.2514/1.J052183.

[137] S. Chen, W. Chen, and S. Lee. Level set based robust shape and topology optimization under random field uncertainties. *Structural and Multidisciplinary Optimization*, 41(4):507 – 524, 2010. ISSN 1615-1488. doi:https://doi.org/10.1007/s00158-009-0449-2.

[138] M. Tootkaboni, A. Asadpoure, and J. K. Guest. Topology optimization of continuum structures under uncertainty âĂŞ A Polynomial Chaos approach. *Computer Methods in Applied Mechanics and Engineering*, 201-204:263 – 275, 2012. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2011.09.009. URL `http://www.sciencedirect.com/science/article/pii/S0045782511002982`.

[139] J. K. Guest and T. Igusa. Structural optimization under uncertain loads and nodal locations. *Computer Methods in Applied Mechanics and Engineering*, 198(1):116 – 124, 2008. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2008.04.009. URL `http://www.sciencedirect.com/science/article/pii/S004578250800159X`. Computational Methods in Optimization Considering Uncertainties.

[140] K. Maute. *Topology Optimization under Uncertainty.* Springer Vienna, 2014. ISBN 978-3-7091-1643-2. doi:10.1007/978-3-7091-1643-2_20. URL `https://doi.org/10.1007/978-3-7091-1643-2_20`.

[141] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. Mavriplis. CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. Technical Report NASA/CRâĂŞ2014-218178, NASA Langley Research Center, 2014.