UNIVERSITY OF CALIFORNIA SAN DIEGO

Graph-accelerated uncertainty propagation for large-scale multidisciplinary design, analysis, and optimization under uncertainty

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Engineering Sciences (Aerospace Engineering)

by

Bingran Wang

Committee in charge:

Professor John T. Hwang, Chair
Professor Thomas R. Bewley
Professor Boris M. Kramer
Professor Michael D. Todd

2024

The Dissertation of Bingran Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

DEDICATION

To my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am deeply grateful to have had the opportunity to work under the guidance of Professor John Hwang, whose significant impact on both my personal and professional life throughout my Ph.D. journey cannot be overstated. His unwavering support, patience, integrity, and objective thinking have not only reshaped my attitude towards research but have also greatly influenced my approach to personal development. Under his guidance, I have not only gained valuable research insights but have also learned important life lessons that extend far beyond academia.

I would also like to express my appreciation to the other dissertation committee members, namely, Prof. Thomas Bewley, Prof. Boris Kramer, and Prof. Michael Todd. Their expertise, unwavering support, insightful feedback, and constructive criticism have played an invaluable role in guiding me through the completion of my dissertation.

Finally, I extend my deepest appreciation to my family. I am profoundly grateful to my parents for their unwavering support and unconditional love throughout my life. Their guidance and encouragement have been instrumental in shaping who I am today. Additionally, I would like to express my gratitude to my partner, Trista. Her constant companionship and belief in my abilities have been a source of strength and motivation, inspiring me to overcome challenges and pursue higher goals.

The material presented in this dissertation is, in part, based upon work supported by DARPA under grant No. D23AP00028-00 and by NASA under award No. 80NSSC21M0070.

Chapter 3, is taken, in part, from the material [104] which has bee published as Wang, B., Sperry, M., Gandarillas, V. E., and Hwang, J. T., "Accelerating model evaluations in uncertainty propagation on tensor grids using computational graph transformations," Aerospace Science and Technology, vol. 145, 2024, pp. 108843. The dissertation author was the primary investigator and author of this chapter.

Chapter 4, is taken, in part, from the material [98] which has been submitted for publication as Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J. T., "Graph-accelerated non-intrusive polynomial chaos expansion using partially tensor-structured quadrature rules," Aerospace Sci-

ence and Technology. The dissertation author was the primary investigator and author of this paper.

Chapter 5, is taken, in part, from the material [101] which has been submitted for publication as Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J.T., "Extension of graph-accelerated non-intrusive polynomial chaos to high-dimensional uncertainty quantification through the active subspace method," Structural and Multidisciplinary Optimization. The dissertation author was the primary investigator and author of this paper.

Chapter 6, is taken, in part, from the material [102] which has been submitted for publication as Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J.T., "A gradient-enhanced univariate dimension reduction method for uncertainty propagation," Aerospace Science and Technology. The dissertation author was the primary investigator and author of this paper.

Chapter 7, in part, is a reprint of the material [99] as it appears in Wang, B., Orndorff, N. C., Joshy, A. J., and Hwang, J. T., "Graph-accelerated large-scale multidisciplinary design optimization under uncertainty of a laser-beam-powered aircraft," AIAA SciTech 2024 Forum, 2024, p. 0169. The dissertation author was the primary investigator and author of this paper.

VITA

2019	Bachelor of Applied Science in Engineering Science, University of Toronto
2021	Master of Science in Engineering Science (Aerospace Engineering), University of California, San Diego
2024	Doctor of Philosophy in Engineering Science (Aerospace Engineering), University of California, San Diego

PUBLICATIONS

Journal publications

[J5] Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J.T., "Extension of graph-accelerated non-intrusive polynomial chaos to high-dimensional uncertainty quantification through the active subspace method," *Structural and Multidisciplinary Optimization*, submitted.

[J4] Wang, B., Orndorff, N. C., and Hwang, J.T., "Graph-accelerated non-intrusive polynomial chaos expansion using partially tensor-structured quadrature rules," *Aerospace Science and Technology*, submitted.

[J3] Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J. T., "A gradient-enhanced univariate dimension reduction method for uncertainty propagation," *Aerospace Science and Technology*, submitted.

[J2] Wang, B., Sperry, M., Gandarillas, V. E., and Hwang, J. T., "Accelerating model evaluations in uncertainty propagation on tensor grids using computational graph transformations," *Aerospace Science and Technology,* vol. 145, 2024, pp. 108843.

[J1] Wang, B., Joshy, A. J., and Hwang, J. T., "Equality-constrained engineering design optimization using a novel inexact quasi-newton method," *AIAA Journal*, vol. 60, no. 11, 2022, pp. 6157-6167.

Conference papers

[C7] Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J. T., "A gradient-enhanced univariate dimension reduction method for uncertainty propagation." *AIAA SciTech 2024 Forum,* 2024, p. 1234.

[C6] Wang, B., Orndorff, N. C., Joshy, A. J., and Hwang, J. T., "Graph-accelerated large-scale multidisciplinary design optimization under uncertainty of a laser-beam-powered aircraft," *AIAA SciTech 2024 Forum,* 2024, p. 0169.

[C5] Wang, B., Orndorff, N.C., Sperry, M., and Hwang, J. T., "High-dimensional uncertainty quantification using graph-accelerated non-intrusive polynomial chaos and active subspace methods," *AIAA Aviation 2023 Forum,* 2023, p. 4264.

[C4] Orndorff, N. C., Wang, B., Ruh, M. L., Fletcher, A., and Hwang, J. T., "Gradient-based sizing optimization of power-beaming-enabled aircraft." *AIAA Aviation 2023 Forum,* 2023, p. 4019.

[C3] Wang, B., Orndorff, N. C., and Hwang, J. T., "Optimally tensor-structured quadrature rule for uncertainty quantification," *AIAA SciTech 2023 Forum,* 2023, p. 0741.

[C2] Wang, B., Sperry, M., Gandarillas, V. E., and Hwang, J. T., "Efficient uncertainty propagation through computational graph modification and automatic code generation," *AIAA Aviation 2022 Forum,* 2022, p. 3997.

[C1] Wang, B., Joshy, A. J., and Hwang, J. T., "An adaptive, inexact gradient-based algorithm for multidisciplinary design optimization," *AIAA Aviation 2021 Forum,* 2021, p. 3041.

ABSTRACT OF THE DISSERTATION

Graph-accelerated uncertainty propagation for large-scale multidisciplinary design, analysis, and optimization under uncertainty

by

Bingran Wang

Doctor of Philosophy in Engineering Sciences (Aerospace Engineering)

University of California San Diego, 2024

Professor John T. Hwang, Chair

Aerospace vehicle design is a complex process involving multiple engineering disciplines such as aerodynamics, structures, propulsion, and controls. This complexity led to the emergence of multidisciplinary design optimization (MDO), a field that employs numerical optimization methods to improve the designs of engineering systems while simultaneously considering multiple disciplines. Integrating all disciplines significantly increases the problem's complexity, which motivates the use of gradient-based optimization methods with analytical computation of the derivatives. This approach has demonstrated considerable success in tackling practical large-scale MDO problems with hundreds of design variables. Recent applications can be found

in the conceptual design of electrified aircraft, unmanned aerial vehicles, and launch vehicles.

Presently, most MDO applications formulate and solve deterministic optimization problems in which the objective and constraint functions do not consider randomness. However, real-world scenarios introduce uncertainties arising from factors like variations in operating conditions, parameter fluctuations, and manufacturing discrepancies. Incorporating these uncertainties into MDO problems transforms them into MDO under uncertainty (MDOUU) problems. Addressing well-defined MDOUU problems can enhance the robustness and reliability of MDO-optimized designs in real-world scenarios.

This dissertation presents a suite of graph-accelerated uncertainty propagation methods, designed to tackle various forward uncertainty quantification (UQ) and MDOUU problems. At the core of these methods lies a new computational graph transformation method called *Accelerated Model Evaluations on Tensor Grids using Computational Graph Transformations* (AMTC). AMTC leverages the sparsity in the computational graphs of multidisciplinary systems to accelerate tensor-grid evaluations. This approach has been effectively combined with the non-intrusive polynomial chaos method to tackle UQ and MDOUU problems, with several extensions devised to address higher-dimensional problems within this framework. Towards the end, this dissertation also includes a case study on a laser-beam-powered aircraft design problem. This study offers a comparative analysis between the results of MDO and MDOUU, while also demonstrating the efficacy of graph-accelerated uncertainty propagation methods in addressing large-scale MDOUU problems.

Chapter 1

Introduction

The primary goal of this thesis is to devise efficient uncertainty propagation methods that seamlessly integrate into the state-of-the-art multidisciplinary design optimization (MDO) framework, thereby enhancing the practicality of MDO-optimized designs. Multidisciplinary design optimization (MDO) has gained popularity in both industry and academia for its success in finding improved designs in practical engineering problems by using numerical optimization techniques. However, a significant challenge lies in the absence of computationally efficient uncertainty-based methods that can be seamlessly integrated into the existing gradient-based MDO framework. This thesis tackles this challenge by proposing a suite of graph-accelerated uncertainty propagation methods. These methods capitalize on the sparsity inherent in the computational graphs of multidisciplinary systems and are designed for seamless integration into the gradient-based MDO frameworks.

1.1 Multidisciplinary design analysis and optimization

The origin of multidisciplinary design Optimization (MDO), also known as multidisciplinary design analysis and optimization (MDAO), can be traced back to the 1960s. In 1960, Schmit coupled numerical optimization methods with structural computational models to optimize structural designs [86], which led to the emergence of the structural optimization field. In 1965, Schmit and Thornton demonstrated the first MDO application in which they optimized the

airfoil design considering both structural and aerodynamic models [87].

Since then, structural optimization gradually expanded to include shape optimization of 3-dimensional structures [34], and topology optimization of material layout [21]. Concurrently, aerodynamic shape optimization emerged from Pironneau's utilization of optimal control techniques for optimizing airfoil shapes [77]. Jameson further extended this framework by integrating the adjoint method with computational fluid dynamics (CFD) models to optimize aircraft wings [43].

Initially, MDO applications were focused on coupling aerodynamics and structures in wing design. However, this has evolved to encompass other disciplines like propulsion and controls, leading to the use of MDO in aircraft configuration designs. The development of MDO methodology has seen efforts towards distributed MDO architectures [65, 48], analytical derivative computations [63, 66, 41, 92], and optimization algorithms [36, 96, 47]. These advancements have facilitated the solution of large-scale MDO problems and the development of MDO software tools like OpenMDAO [32], SU2 [20], and CSDL [26].

With the continuous development and implementation of MDO methods in software, the impact of MDO has expanded significantly across various applications. This includes improving the design of electrified aircraft [4, 13, 84, 80], supersonic aircraft [64, 88], unmanned aerial vehicles [51, 72], launch vehicles [6, 14], wind turbines [53, 69], satellites [45, 40], and many other engineering systems. A comprehensive MDO textbook by Martins and Ning can be found in [65].

1.2 Motivation for MDO under uncertainty (MDOUU)

In a realistic world, aerospace vehicle systems are inherently subject to various uncertainties arising from both the system itself and the environmental and operational conditions it encounters. For example, in aircraft design, uncertainties include fluctuations in operating conditions, prediction errors due to design model assumptions and simplifications, and discrepancies

associated with manufacturing tolerances. These uncertainties can cause system performance to change or fluctuate, potentially leading to severe deviations, unanticipated functional faults, and even mission failures. Therefore, it is crucial to account for uncertainties from the beginning of the aerospace vehicle system design process. When we incorporate uncertainties into the MDO problems, it transforms the MDO problems into the MDO under uncertainty (MDOUU) problems.

Computational methods that incorporate uncertainties are typically described as non-deterministic approaches. These approaches aim to address two main issues: (1) improving the robustness of aerospace vehicles to decrease their sensitivity to variations and (2) enhancing the reliability of aerospace vehicles to decrease the likelihood of functional failure under potential critical conditions. Corresponding to these two design aims, MDOUU problems can be classified into two primary categories: robust design optimization (RDO) and reliability-based design optimization (RBDO). Robust design optimization seeks the optimal design that is less sensitive to variations in uncertain inputs, thereby achieving stable performance. In contrast, reliability-based design optimization seeks the optimal design with guarantees of reliability, ensuring the system performs correctly under specified critical conditions.

Optimization under uncertainty has its origins in the 1950s [19, 25] and has since experienced significant growth. A substantial body of research has been devoted to developing optimization algorithms that account for uncertainty, leading to successful applications in various fields. This is particularly evident in aerospace engineering, which has strict requirements for system reliability and robustness. In aerospace engineering, research on design optimization under uncertainty has primarily focused on single disciplines such as structures [59, 11] and aerodynamics [85, 33]. MDOUU gained popularity in the 2000s [73, 3]. By systematically considering uncertainties in the multidisciplinary system, MDOUU can result in optimal designs that are more robust and reliable to operate in real-world scenarios. In 2002, NASA published a white paper [110], which addresses the needs and opportunities for uncertainty-based multidisciplinary design for aerospace vehicles. This white paper also highlights that the lack of efficient

computational methods is one of the main obstacles to applying uncertainty-based methods to practical aerospace vehicle design problems.

The development of MDOUU methodology has seen significant efforts towards uncertainty propagation methods [28, 75], surrogate and approximation methods [8], and optimization under uncertainty algorithms [83, 82]. Recent applications of MDOUU in aerospace engineering can be found in [68, 42]. A recent review paper on uncertainty-based MDO is available in [109]. Additionally, a comprehensive textbook on aerospace systems analysis and optimization under uncertainty by Brevault et al. can be found in [15].

1.3 Thesis roadmap

This dissertation is organized as follows:

- Chapter 2 presents related background information for this thesis.
- Chapter 3 presents a computational graph transformation method called *Accelerated Model Evaluations on Tensor Grids using Computational Graph Transformation* (AMTC). This method reduces tensor-grid evaluations by leveraging the sparsity within the model's computational graph. When combined with the integration-based non-intrusive polynomial chaos (NIPC) method, AMTC efficiently addresses many low-dimensional (fewer than four dimensions) uncertainty quantification (UQ) problems in multidisciplinary systems. This integrated framework is referred to as the graph-accelerated NIPC method.
- Chapter 4 introduces a framework for generating a partially tensor-structured quadrature rule tailored for use with AMTC. This framework generates the quadrature rule using the designed method and identifies a desired tensor structure by analyzing the model's computational graph. By doing so, it broadens the scope of the graph-accelerated NIPC methods, making them applicable to a wider range of multidisciplinary UQ problems with higher dimensions (up to 10 dimensions).

- Chapter 5 presents two methods, AS-NIPC and AS-AMTC. AS-NIPC combines the integration-based NIPC with the AS method for solving high-dimensional UQ problems. AS-NIPC incorporates rigorous approaches to generate orthogonal polynomial basis functions for lower-dimensional active variables and efficient quadrature rules to estimate their coefficients. Building upon AS-NIPC, the AS-AMTC method extends the integration of the active subspace method with graph-accelerated NIPC methods. This extension enhances the applicability of graph-accelerated NIPC methods to even higher dimensions, exceeding 10 dimensions, thus broadening their utility in addressing high-dimensional multidisciplinary UQ problems.

- Chapter 6 presents a UQ method, gradient-enhanced univariate dimension reduction (GUDR). This method enhances on the univariate dimension reduction (UDR) method by incorporating univariate gradient terms. The GUDR method relies on the AMTC method to achieve efficient tensor-grid evaluations and the acceleration of AMTC does not rely on the computational graph sparsity of the original computational model. Notably, with the use of an efficient automatic differentiation method, GUDR preserves the linear scalability of the UDR method while achieving significantly higher accuracy in estimating the higher-order statistical moments.

- Chapter 7 presents a case study on a laser-beam-powered aircraft design problem. This study offers a comparative analysis between the results of MDO and MDOUU, while also demonstrating the efficacy of graph-accelerated uncertainty propagation methods in addressing large-scale MDOUU problems.

- Chapter 8 offers the summary of contributions and recommendations for future work.

Chapter 2

Background

2.1 MDO formulation and framework

The mathematical problem of a deterministic optimization problem can be expressed as follows:

$$\begin{aligned} \min_{x} \quad & \mathscr{F}(x) \\ \text{subject to} \quad & \mathscr{C}(x) \leq 0 \\ & x^L \geq x \leq x^U, \end{aligned} \tag{2.1}$$

where x is the vector of design variables; x^L and x^U are the lower and upper bounds of the design variables, respectively; $\mathscr{F}$ denotes the objective function; and $\mathscr{C}$ corresponds to the vector-valued constraint function. In MDO, optimization problems often feature nonlinear functions, high-dimensional design spaces, and expensive model evaluations due to the involvement of multiple disciplines. Consequently, iterative gradient-based optimization methods, such as the quasi-Newton method, are commonly employed for their ability to efficiently find local minima with excellent scalability in high-dimensional design spaces. The commonly used optimizers include SNOPT [29] and SciPy's SLSQP, which use sequential quadratic programming (SQP) methods to tackle high-dimensional constrained optimization problems with smooth nonlinear functions. In this framework, the optimizer updates the design variables, x in each optimization iteration using the function and gradient evaluations of the objective and constraint functions until the optimizer converges. The computational framework for gradient-based MDO is shown

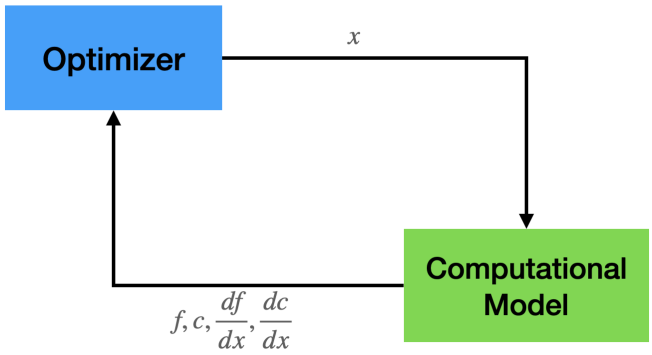


Figure 2.1. Computational framework for gradient-based MDO

in Fig. 2.1.

2.2 MDOUU formulation and framework

In this thesis, we consider the MDOUU problems within a probabilistic framework, in which uncertain inputs are defined as random variables with continuous probability density distributions. Additionally, we assume the uncertain inputs are independent of the design variables. In this context, when uncertain inputs are introduced into this optimization framework, the outputs of the objective and constraint functions become random variables, rendering the optimization formulation in (2.1) invalid. In MDOUU, there are two common ways to formulate the optimization problems: robust design optimization (RDO) and reliability-based design optimization (RBDO). In RDO, the optimization problem can be formulated as:

$$\begin{aligned} \min_{x} \quad & \mathscr{M}(x) := E[f(x,U)] + \alpha_1 S[f(x,U)] \\ \text{subject to} \quad & \mathscr{N}(x) := E[c(x,U)] + \alpha_2 S[c(x,U)] \leq 0 \\ & x^L \leq x \leq x^U. \end{aligned} \tag{2.2}$$

Here, U denotes the vector of uncertain inputs; $f(x,U)$ and $c(x,U)$ denote the stochastic outputs associated with the MDO objective and constraint functions, respectively; $E[\cdot]$ and $S[\cdot]$ denote the expectation and standard deviation of a random variable, respectively; $\mathscr{M}$ and $\mathscr{N}$ represent

the new MDOUU objective and constraint functions, respectively. These new objective and constraint functions incorporate the statistical moments of the original functions, and α_1 and α_2 are tunable parameters associated with the penalization of standard deviations, respectively. This formulation is often easy to implement and has proven to efficiently improve the robustness of designs under input variations.

In RBDO, the optimization problem can be formulated as:

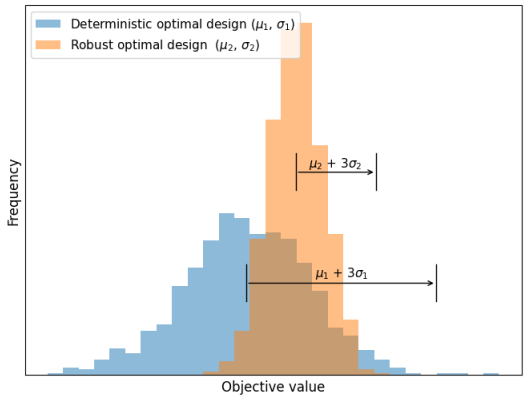
$$\begin{aligned} \min_{x} \quad & \mathscr{M}(x) := E[\mathscr{F}(x,U)] \\ \text{subject to} \quad & \mathscr{N}(x) := P[\mathscr{C}(x,U) \leq 0] \leq p \\ & x^L \geq x \leq x^U \end{aligned} \tag{2.3}$$

where P is the probability function and p is the required probability of failure. In RBDO, the primary objective is to find the optimal design that satisfies the probability of failure constraints, thereby achieving optimal performance with reliability guarantees. A demonstration of RDO and RBDO results is shown in Fig. 2.2 . The figure includes a comparison between the objective random outputs of RDO and deterministic optimization results, as well as a comparison between the constraint random outputs of RBDO and deterministic optimization results.

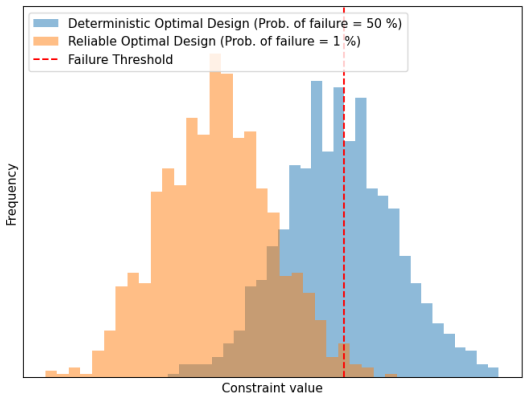
Solving the MDOUU problems typically requires a double-loop framework with optimization in the outer loop and uncertainty quantification (UQ) in the inner loop. When using a gradient-based optimizer, at each optimization iteration, we use a UQ solver to solve $\mathscr{M}$ and $\mathscr{N}$ and their gradients with respect to the design variables, $\frac{d\mathscr{M}}{dx}$ and $\frac{d\mathscr{N}}{dx}$. These information is passed into the optimizer to update the design variables. The computational framework for gradient-based MDOUU is shown in Fig. 2.3.

2.3 Uncertainty propagation methods

Due to the double-loop structure of the computational framework for MDOUU, solving MDOUU is often significantly more expensive than solving the corresponding MDO problem.



(a) RDO result vs deterministic optimization result



(b) RBDO result vs deterministic optimization result

Figure 2.2. Comparison of RDO, RBDO, and deterministic optimization results

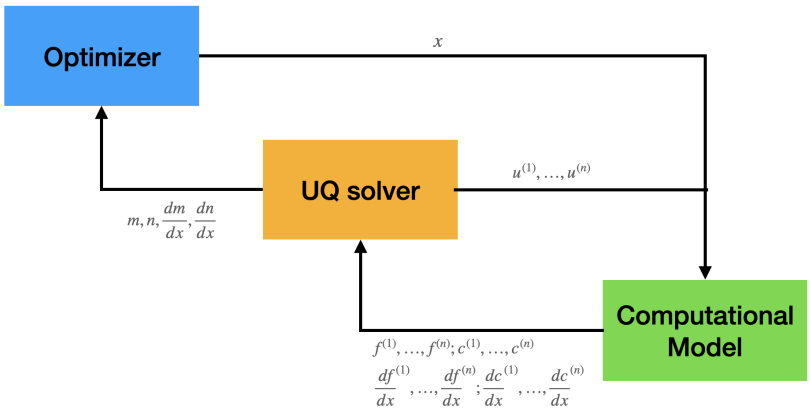


Figure 2.3. Computational framework for gradient-based MDOUU

One of the key research directions to solve large-scale MDOUU problems is to devise efficient UQ methods to minimize the UQ cost in each optimization iteration. In MDOUU, the UQ problem is a forward problem which is also known as uncertainty propagation. In practical scenarios, numerical models inherently fall short of achieving perfect fidelity with reality and consistently contend with inherent uncertainties. To address this challenge, uncertainty propagation aims to rigorously examine and analyze how a system's output is influenced by the uncertainties inherent in its inputs. The resultant assessments of uncertainty serve as a valuable foundation for informed decision-making and thorough risk assessment. Applications of UQ can be found in many scientific and engineering domains, such as weather forecasting [49, 74], structural analysis [95, 39], and aircraft design [68, 100, 58].

Uncertainties within the inputs can stem from diverse origins and are typically classified into two categories: aleatoric and epistemic. Aleatoric uncertainties, recognized as irreducible uncertainties, are intrinsic to the system and encompass factors such as fluctuations in operational conditions, mission prerequisites, and model parameters. These uncertainties can often be described within a probabilistic framework. In contrast, epistemic uncertainties constitute reducible uncertainties arising from knowledge gaps. These uncertainties may be induced by approximations utilized in computational models or numerical solution methods, and are often difficult to be described within a probabilistic framework. This thesis focuses on UQ problems within a probabilistic formalism where uncertain inputs are continuous random variables with known probability density distributions. The goal is to estimate statistical moments such as the mean and variance of the Quantity of Interest (QoI), or more complex risk measures such as the probability of failure or the conditional value at risk.

For this type of UQ problem, the most common non-intrusive methods include the method of moments, polynomial chaos, kriging, and Monte Carlo. The method of moments analytically computes the statistical moments of the QoI, using the Taylor series approximation. It often only uses function evaluations and derivative information at one or a small number of input points [106]. The most commonly employed variants of this method utilize derivatives up to

either first [24] or second order [60, 61] to ensure cost-effectiveness. While adept at efficiently estimating statistical moments, this approach grapples with challenges in estimating other kinds of risk measures. Moreover, it may lack accuracy when input variance is significant. In contrast, the Monte Carlo approach employs random sampling of uncertain inputs to compute the risk measures. Based on the law of large numbers, Monte Carlo's convergence rate is independent of the number of uncertain inputs, making it particularly attractive for solving high-dimensional problems. Recent strides in enhancing its efficiency encompass multi-fidelity [75, 76] and importance sampling [93] techniques. However, in the context of low-dimensional problems, the Monte Carlo method might demand substantially more model evaluations to achieve the same level of accuracy as the alternative UQ methodologies. Kriging, also known as Gaussian process regression, is a common statistical technique employed in various fields. In UQ, kriging constructs a surrogate response surface by leveraging input-output data pairs. The surrogate model replaces the original expensive function to allow a large number of model evaluations in order to perform reliability analysis [52, 38] or optimization under uncertainty [82]. For low-dimensional UQ problems, if the objective function is assumed to be smooth, the polynomial chaos-based techniques often stand out as the most efficacious option. The generalized polynomial chaos theory represents the QoI as orthogonal polynomials that are derived from the distributions of the uncertain inputs. Benefiting from the inherent smoothness of the random space, polynomial chaos-based methods exhibit rapid convergence rates facilitated by sampling or integration techniques. Common polynomial chaos-based methods include non-intrusive polynomial chaos (NIPC) [37, 46, 54] and stochastic collocation (SC) [107, 5].

2.3.1 Polynomial chaos expansion

Wiener initially introduced the concept of polynomial chaos expansion (PCE), utilizing Hermite polynomials to model the response of a system affected by Gaussian uncertain inputs [105]. This foundational concept was later advanced into the generalized polynomial chaos (gPC) method by Xiu and Karniadakis [108]. Within the framework of gPC, model responses are

represented through polynomial series of uncertain inputs. These polynomials are meticulously chosen to exhibit orthogonality with respect to the probability distribution of the uncertain inputs, thereby ensuring exponential convergence.

We address a general problem in uncertainty quantification (UQ) involving a function defined as follows:

$$f = \mathscr{F}(u), \tag{2.4}$$

where $\mathscr{F} : R^d \to R$ represents the model evaluation function, $u \in R^d$ denotes the input vector, and $f \in R$ represents a scalar output. The uncertainties associated with the inputs are expressed as a stochastic vector denoted by $U := [U_1, \ldots, U_d]$, under the assumption that these random variables are mutually independent. The stochastic input vector adheres to the probability distribution $\rho(u)$ with its support defined by Γ. Our focus centers on the evaluation of risk measures of the output random variable, $f(U)$.

In gPC, the stochastic output, $f(U)$, is defined as an infinite series of orthogonal polynomials with respect to the uncertain input variables:

$$f(U) = \sum_{i=0}^{\infty} \alpha_i \Phi_i(U), \tag{2.5}$$

where $\Phi_i(U)$ are the PCE basis functions that are generated based on $\rho(U)$, and α_i are the corresponding weights that need to be determined.

Choosing the PCE basis functions is important to ensure the rapid convergence of PCE-based methods. These PCE basis functions have to satisfy the orthogonality property,

$$\left\langle \Phi_i(U), \Phi_j(U) \right\rangle = \delta_{ij}, \tag{2.6}$$

where δ_{ij} is Kronecker delta and the inner product is defined as

$$\left\langle \Phi_i(U), \Phi_j(U) \right\rangle = \int_{\Gamma} \Phi_i(u) \Phi_j(u) \rho(u) du. \tag{2.7}$$

In one-dimensional problems, we can directly use the univariate orthogonal polynomials as PCE basis functions. In Table 2.1, we show the univariate orthogonal polynomials for common types of continuous random variables. In multi-dimensional problems, if the input variables are mutually independent, the PCE basis functions can be formed as a tensor-product of the univariate orthogonal polynomials corresponding to each uncertain input. To elucidate this, we introduce the concept of multi-index notation and define a p-tuple as

$$i' = (i_1, \ldots, i_p) \in N_0^p, \tag{2.8}$$

with its magnitude determined as

$$|i'| = i_1 + \ldots + i_p. \tag{2.9}$$

Employing this notation, we denote the collection of univariate PCE basis functions for variable u_k as $\{\phi_i(u_k)\}_{i=0}^{p}$. Then, the multivariate PCE basis with an upper limit on the total degree of p can be expressed as

$$\Phi_{i'}(u) = \phi_{i_1}(u_1) \ldots \phi_{i_d}(u_d), \quad |i'| \leq p. \tag{2.10}$$

In practice, one may use the PCE basis functions in (2.10) to truncate the infinite series in (2.5), resulting in:

$$f(U) \approx \sum_{i=0}^{q} \alpha_i \Phi_i(U). \tag{2.11}$$

The resultant number of PCE basis functions, $q+1$, satisfies:

$$q+1 = \frac{(d+p)!}{d!p!}. \tag{2.12}$$

Once the PCE coefficients are estimated, it becomes straightforward to compute statistical moments such as the mean and standard deviation of the model output. These calculations can be expressed as follows:

$$\mu_f = \alpha_0, \tag{2.13}$$

Table 2.1. Orthogonal polynomials for common types of continuous random variables

Distribution	Orthogonal polynomials	Support range
Normal	Hermite	$(-\infty,\infty)$
Uniform	Legendre	$[-1,1]$
Exponential	Laguerre	$[0,\infty)$
Beta	Jacobi	$(-1,1)$
Gamma	Generalized Laguerre	$[0,\infty)$

$$\sigma_f = \sum_{i=1}^{d} \alpha_i^2. \tag{2.14}$$

For risk measures like the probability of failure, the truncated PCE model can be treated as a surrogate model, and methods like Monte Carlo can be applied on the surrogate model to compute the desired risk measure.

2.3.2 Non-intrusive polynomial chaos method

The non-intrusive polynomial chaos (NIPC) method solves the PCE coefficients α_i in (2.11) by either integration or regression. The integration approach uses the orthogonal property in (2.6) and projects the model output onto each basis function:

$$\alpha_i = \frac{\langle f(U), \Phi_i \rangle}{\langle \Phi_i^2 \rangle} = \frac{1}{\langle \Phi_i^2 \rangle} \int_u f(u)\Phi_i(u)\rho(u)du. \tag{2.15}$$

This requires solving a multi-dimensional integration problem, which is often estimated by using the Gauss quadrature approach:

$$\begin{aligned} \alpha_i &= \frac{1}{\langle \Phi_i^2 \rangle} \int_{u_1} \dots \int_{u_d} f(u_1,\dots,u_d)\Phi_k(u_1,\dots,u_d)\rho(u_1,\dots,u_d)du_1\dots du_d \\ &\approx \frac{1}{\langle \Phi_i^2 \rangle} \sum_{i_1=0}^{k} \dots \sum_{i_d=0}^{k} w_1^{(i_1)} \dots w_d^{(i_d)} f(u_1^{(i_1)},\dots,u_d^{(i_d)})\Phi_i(u_1^{(i_1)},\dots,u_d^{(i_d)}), \end{aligned} \tag{2.16}$$

where $(u_i^{(1)},\dots,u_i^{(k)})$ and $(w_i^{(1)},\dots,w_i^{(k)})$ are the nodes and weights for the 1D Gauss quadrature rule in the u_i dimension, k is the number of quadrature points used in each dimension. The 1D

quadrature rule with k nodes is chosen such that the quadrature rule can exactly integrate 1D polynomials up to $(2k-1)$th order. In the multi-dimensional case, the Gauss quadrature rule forms a tensor product of the 1D quadrature rule, which requires us to evaluate the model output at the tensor-product quadrature points, defined as

$$u = u_1^k \times \cdots \times u_d^k, \tag{2.17}$$

where

$$u_i^k := \{u_i^{(j)}\}_{j=1}^k. \tag{2.18}$$

The total number of quadrature points is thus k^d, and this number increases exponentially with respect to the number of uncertain input variables. For high-dimensional problems, one way to mitigate the exponential growth of quadrature points is to use the Smolyak sparse grid approach [70], which drops the higher-order cross terms in the quadrature points with minimal loss of accuracy. The sparse grid quadrature points can be expressed as:

$$u = \bigcup_{\ell-d+1 \le |i| \le \ell} \left(u_1^{i_1} \times \cdots \times u_d^{i_d} \right), \tag{2.19}$$

where ℓ is the level of construction. The existing variations of sparse-grid methods can be found in [27]. Another way to reduce the total number of quadrature points is to use a designed quadrature approach. In [55], Keshavarzzadeh et al. formulate an optimization problem to optimize for the designed quadrature points and the corresponding weights that ensure the exact integration on a polynomial subspace. The number of designed quadrature points can be significantly smaller than the full-grid quadrature points for solving the high-dimensional integration problems for the same level of accuracy.

The regression approach first generates n sample points for U and evaluates the model

for each sample point. The coefficients are determined by solving a linear least-squares problem:

$$\begin{bmatrix} \Phi_1(u^{(1)}) & \dots & \Phi_q(u^{(1)}) \\ \vdots & & \vdots \\ \Phi_1(u^{(n)}) & \dots & \Phi_q(u^{(n)}) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_q \end{bmatrix} = \begin{bmatrix} f(u^{(1)}) \\ \vdots \\ f(u^{(n)}) \end{bmatrix}. \tag{2.20}$$

The rule of thumb is that the number of samples n needs to be 2-3 times the number of coefficients q. This means the cost of the regression-based NIPC method can be reduced by using the sparse PCE basis, resulting in fewer coefficients to be estimated. Recent advances focus on adaptive basis selection and adaptive sampling[10, 94, 62].

Chapter 3

Computational graph transformations to accelerate tensor-grid evaluations in uncertainty quantification (UQ)

For low-dimensional uncertainty quantification (UQ) problems, polynomial chaos-based methods are often the most effective. When using non-intrusive polynomial chaos (NIPC) and stochastic collocation (SC) methods, a tensor-product grid can be used to sample the input space. However, this approach is typically reserved for very low-dimensional UQ problems due to the exponential increase in input points with the number of uncertain inputs. More commonly, methods employ sparse grids or randomly generated sample points to reduce the number of input points while minimizing the loss of accuracy. Our findings, however, show that maintaining a tensor structure for the input points offers unique advantages. For certain problems, the total model evaluation cost can be reduced through a different approach without altering the tensor structure of the input points.

When evaluating a computational model on full-grid input points, if we view the computational model as a computational graph with only elementary operations, the current framework evaluates each operation the same number of times at the tensor-product input points of all the uncertain inputs. However, for each operation, the output data only has distinct values at the distinct input points in their dependent input space, and a large portion of the operations may not be dependent on all of the uncertain inputs. This means the current framework of NIPC and SC

could create many wasteful evaluations on a large portion of the operations.

This chapter presents a computational graph transformation method called *Accelerated Model evaluations on Tensor grids using Computational graph transformations* (AMTC) that algorithmically modifies the computational graph to eliminate the unnecessary evaluations incurred by the current framework of integration-based NIPC and SC. AMTC reduces the model evaluation cost on full-grid quadrature points by partitioning the computational graph into sub-graphs using the operations' dependency information and evaluating each sub-graph on the distinct quadrature points. The proposed method is implemented in conjunction with a new algebraic modeling language, the Computational System Design Language (CSDL). We achieve cost reductions on UQ problems in a general and automatic way by analyzing and manipulating the computational graph that CSDL makes available.

This method has been applied to four different UQ problems with different computational graph structures. For three of the problems, we observed a significant acceleration in model evaluation time when solving the UQ problems using the full-grid NIPC method, which makes this UQ method the most efficient method among the implemented UQ methods for these problems. Generally, for a wide range of UQ problems that involve multi-point, multi-disciplinary, or other models that possess a sparse graph structure, we expect this method to significantly reduce the model evaluation time on tensor-grid input points, which can improve the time scaling for, but not limited to NIPC and SC methods.

3.1 Model evaluation cost for current non-intrusive UQ methods

If we consider only the non-adaptive approaches, the current framework for non-intrusive UQ methods includes three steps:

1. Generate sample/quadrature points based on the distribution of uncertain inputs.

2. Evaluate the target model for each of the sample points.

3. Post-process the output points to compute the desired quantities of the output.

Step one generates the input points based on the distribution of uncertain inputs. For methods like regression-based NIPC, Monte Carlo and kriging, the input points are often generated randomly (e.g. random sampling, Latin hypercube sampling). For integration-based NIPC and SC methods, they can be generated using the full/sparse-grid strategy. In step two, the target model is repeatedly evaluated at each of the input points. For some models that support vectorized input, the input points can be vectorized and evaluated at the same time. However, most of the time, the target model needs to be run in a for-loop by evaluating one input point at a time. In step three, the model output points are used to estimate the PCE coefficients for NIPC, form Lagrange interpolation functions for SC, or construct Gaussian process model for kriging, before computing the desired quantity for the output.

Under this framework, the model evaluation cost for any UQ method can be approximated as:

$$\text{cost} \approx nO(\mathscr{F}(u)), \tag{3.1}$$

where n is the number of sample points and $O(\mathscr{F}(u))$ is the model evaluation cost for one sample point. When we use the full-grid approach for integration-based NIPC or SC to generate the input points, the number of input points increases exponentially with the dimension of uncertainty inputs as $n = k^d$. Under the current framework to evaluate the model, the evaluation cost follows Eq. 3.1 and thus increases exponentially as the number of uncertain inputs, thus suffers the curse of dimensionality.

The current methods (e.g. sparse-grid and designed quadrature) to overcome the curse of dimensionality all focus on reducing the number of input points, n, by breaking the full-tensor structure of the input points, so that the input points do not increase exponentially as the number of the uncertain inputs, but none tries to break the linear relationship in Eq. 3.1 to reduce the evaluation cost on tensor-grid input points by using a computational graph transformation approach.

3.2 Viewing tensor-grid evaluations via computational graph

Any computer program that describes a computational model is a computational process that executes a sequence of elementary functions and operations. A computational process can be described by a computational graph. A computational graph is a directed acyclic graph in which the nodes represent the operations and variables, and the directed edges represent how operations and variables are connected to each other. Such a graph is also bipartite because each operation is connected to an output variable, and likewise, variables are connected to operations (as arguments). For example, the computer program that evaluates a function $f = cos(u_1) + exp(-u_2)$ can be decomposed into the following set of fundamental operations:

$$\begin{aligned} \xi_1 &= \varphi_1(u_1) = cos(u_1); \\ \xi_2 &= \varphi_2(u_2) = -u_2; \\ \xi_3 &= \varphi_3(\xi_2) = exp(\xi_2); \\ f &= \varphi_4(\xi_1, \xi_3) = \xi_1 + \xi_3, \end{aligned} \tag{3.2}$$

The computational graph for this model is shown in Fig. 3.1 with the blue nodes representing the variables and the orange nodes representing the operations. A common use of computational graphs is in automatic differentiation (AD) [7, 92] to automatically compute the total derivatives. AD views the computer program as a sequence of operations and functions and repeatedly applies the chain rule to compute the total derivative. Another use of the computational graph is to modify the computational graph to reduce the time of memory cost of the model. In Tensorflow [1], a graph optimization method is implemented to reduce memory usage and evaluation costs for neural network models. It modifies the computational graph through constant folding, arithmetic simplification, and optimization.

If we view the model as a computational graph, in the current UQ framework, each

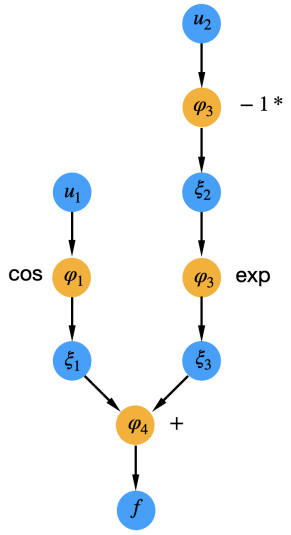


Figure 3.1. Computational graph of function $f = cos(u_1) + exp(-u_2)$

operation and function is evaluated the same number of times as the number of input points. However, this could create many unnecessary evaluations when we evaluate the model on tensor-grid input points. For example, consider a UQ problem involving the simple function we showed before, given by

$$f = \cos(u_1) + \exp(-u_2). \tag{3.3}$$

In this problem, U_1 and U_2 are independent uncertain inputs, and we aim to compute the random variable $f(U_1, U_2)$. The computational process that describes this function is in (4.25). If we choose the full-grid quadrature points with k quadrature points in each dimension, we need to evaluate this function at a total of k^2 quadrature points which are

$$\mathbf{u} = \begin{Bmatrix} (u_1^{(1)}, u_2^{(1)}) & \dots & (u_1^{(k)}, u_2^{(1)}) \\ \vdots & \ddots & \vdots \\ (u_1^{(1)}, u_2^{(k)}) & \dots & (u_1^{(k)}, u_2^{(k)}) \end{Bmatrix}. \tag{3.4}$$

For both inputs u_1 and u_2, if we view their input points as vectors, they are in a tensor-product

form between the quadrature points in its dimension and a vector of ones:

$$
\begin{aligned}
u_1 &= \mathrm{vec}\left(\begin{bmatrix} u_1^{(1)} & \dots & u_1^{(k)} \\ \vdots & \ddots & \vdots \\ u_1^{(1)} & \dots & u_1^{(k)} \end{bmatrix}\right) = \mathrm{vec}\left([u_1^{(1)}, \dots u_1^{(k)}] \otimes [1, \dots, 1]\right), \\
u_2 &= \mathrm{vec}\left(\begin{bmatrix} u_2^{(1)} & \dots & u_2^{(1)} \\ \vdots & \ddots & \vdots \\ u_2^{(k)} & \dots & u_2^{(k)} \end{bmatrix}\right) = \mathrm{vec}\left([1, \dots, 1] \otimes [u_2^{(1)}, \dots u_2^{(k)}]\right).
\end{aligned}
\tag{3.5}
$$

Even though both vectors have a size of k^2, there are only k distinct values, which are the values of the k quadrature points in its dimension. In the current framework, we can view the model evaluation step as propagating the input vectors in (3.5) through the model's computational graph, which is shown in Fig. 3.1. When we view the propagation of input vectors from an operational level, the current framework creates many repeated evaluations. For example, when we evaluate the first operation node in the computational graph, $\xi_1 = \cos(u_1)$, we will evaluate this operation for k^2 points, since there are k^2 input points for u_1. However, out of the k^2 evaluations, there are at most k distinct evaluations and thus the current framework creates $k^2 - k$ wasteful repeated evaluations in this case. Similarly, for $\xi_2 = -1 * u_2$ and $\xi_3 = \exp(\xi_2)$, both of the operations' outputs are only dependent on u_2, and they will also have at most k distinct values to be evaluated for. The current framework also creates $k^2 - k$ repeated evaluations for these operations. In contrast, for the last operation node, $f = \xi_1 + \xi_3$, there are k^2 distinct values to be evaluated as f is dependent on both u_1 and u_2.

In practical UQ problems, we could have a large computational graph with a number of uncertain inputs, but some of the inputs may only affect a small part of the computational graph. For example, for a multidisciplinary model, we may have uncertain inputs coming from different disciplines. When we view the computational graph for the whole model, there may be a small portion of operations that are dependent on some uncertain inputs. In this case, we could reduce

the total computational cost on the tensor-grid inputs by eliminating the repeated evaluation cost for each operation node but the output data still carry the necessary results to compute the results of the QoI.

The logic here is simple and straightforward. One can manually modify the size of the input and state variables in the model to achieve this reduction if the model has a simple graph structure. However, for a complicated model with a large number of uncertainties, it becomes impractical to manually change the code to perform only necessary evaluations and match the shape of input for each node in the computational process.

3.3 Methodology

3.3.1 A computational graph transformation method to accelerate tensor-grid evaluations

Here, we present the *Accelerated Model evaluations on Tensor grids using Computational graph transformations* (AMTC) method. The goal of our method is to enable automatic generation of a modified computational graph, given a computer program, that eliminates all wasteful evaluations due to tensor-product sampling within a NIPC or SC algorithm. The modified computational graph leverages the graph structure of the target model and enables performing the minimal number of evaluations for each operation. To achieve this, we modify the size of input data that are passed into the operations.

In the UQ context, when we evaluate the computational model on full-grid quadrature points, we are forming a tensor product grid from the quadrature points in each uncertain input dimension. If we use k quadrature points in each dimension, we end up having k^d number of input points for d uncertain inputs. For each dimension, these k quadrature points are determined based on the probability distribution of that uncertain input. When we view the target model monolithically, the output is dependent on all of the d uncertain inputs. In the d-dimensional uncertain input space, we have a total of k^d quadrature points, and it is required for us to evaluate

Table 3.1. Operations dependency information for function $f = cos(u_1) + exp(-u_2)$

$D(\varphi_i, u_j)$	u_1	u_2
φ_1	1	0
φ_2	0	1
φ_3	0	1
φ_4	1	1

the model output on all k^d input points to gather all of the necessary information to determine the effect of the uncertain inputs on the model output for UQ purpose. However, when we view it as a computational graph and consider the output of each operation in the graph, not all of the operations' outputs are dependent on d uncertain inputs. For an output dependent on d' uncertain inputs with $d' < d$, in its uncertain input space, there are only $k^{d'}$ number of quadrature points, and accordingly this output only needs to be computed $k^{d'}$ times to compute the necessary information for UQ purpose. Drawing from this rationale, we want to partition the model's computational graph into discrete sub-graphs. These sub-graphs are chosen such that the operations within each are dependent on the same uncertain inputs and thus need to be evaluated on the same quadrature points. We achieve this by generating the dependency information for each operation, which shows whether the operation is dependent on an uncertain input. The dependency information is stored as an *influence matrix*, written as $D(\varphi_i, u_j)$, whose (i, j)th entry is 1 if the ith operation depends on the jth uncertain input and 0 otherwise. For example, the dependency information for the simple function in (4.10) is shown in Tab.4.1.

From the dependency information, the operations that have the same dependency information on all of the uncertain inputs can be grouped together to form a sub-computational graph to be evaluated on the same quadrature points. Each sub-computational graph should be evaluated the same number of times as the number of quadrature points in its uncertain input space. However, the output of one sub-graph may be the input of another sub-graph, since they have different uncertain input spaces, it is required to extend the uncertain input space of that output to match the uncertain input space of the second sub-graph's output. This requires us to perform a deliberate tensor-algebra operation. Fortunately, we can use the Einstein summation

(Einsum) operation to achieve that. The Einsum operation can be used for compactly performing summations involving indices that appear in multiple tensors/matrices. In this case, we can treat the output of the sub-graph as a tensor and use the Einsum operation to perform a summation on specific indices between the output tensor and a tensor of ones, in order to extend its uncertain input space to a desired higher-order input space. For example, consider an addition operation in the computational graph

$$f = \xi_1 + \xi_2$$

with $\xi_1(u_1) \in R^{k_{u_1}}$ and $\xi_2(u_2) \in R^{k_{u_2}}$. In this case, ξ_1 is only dependent on u_1 with a size of k_{u_1}, while ξ_2 is only dependent on u_2 with a size of k_{u_2}. The values in ξ_1 and ξ_2 correspond to their evaluations on the quadrature points in u_1 and u_2 dimensions, respectively. The output of this operation, f is dependent on both u_1 and u_2 and its vector should have a size of $R^{k_{u_1} k_{u_2}}$. What we need to do here is to insert operations to modify the sizes of the inputs so that they have the same size as the output f is supposed to be. Thus we evaluate the addition operation, it results in the correct vector for it. We show a demonstration of how to extend the input size in a Python implementation using NumPy's Einstein summation function followed by a reshape operation. The specific code in Python would be

```
import numpy as np
original_shape_xi_1 = (k_u1,1) #Original size for xi_1
original_shape_xi_2 = (k_u2,1) #Original size for xi_2
modified_shape = (k_u1*k_u2, 1) #Target size for xi_1 and xi_2
xi_1 = np.einsum('i...,p...->pi...', xi_1, np.ones((k_u2, 1)) #(k_u1,1)
                                        -> (k_u1,k_u2,1)
xi_2 = np.einsum('i...,p...->ip...', xi_2, np.ones((k_u1, 1)) #(k_u2,1)
                                        -> (k_u1,k_u2,1)
xi_1 = np.reshape(xi_1, modified_shape) #(k_u1,k_u2,1) ->(k_u1*k_u2, 1)
xi_2 = np.reshape(xi_2, modified_shape) #(k_u1,k_u2,1) ->(k_u1*k_u2, 1).
```

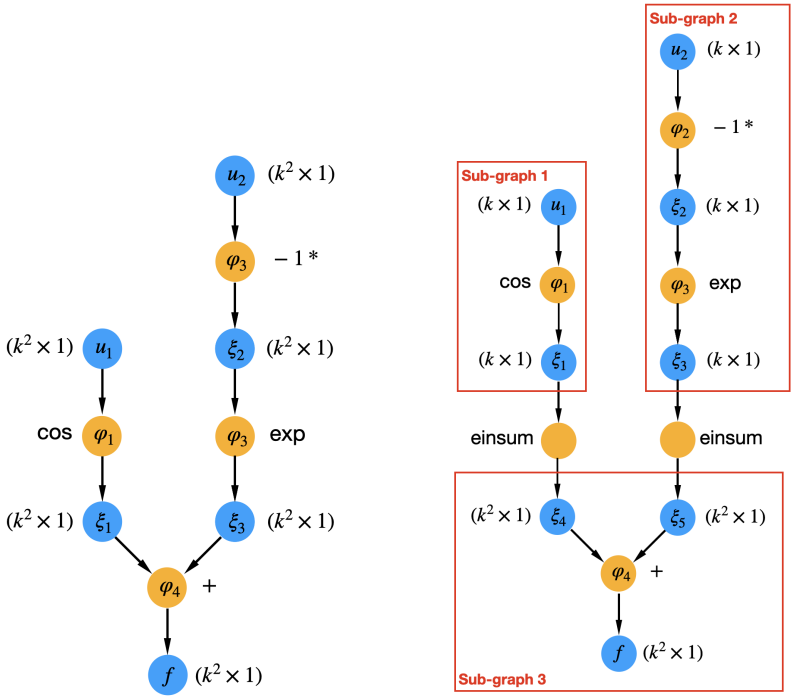
The code above transforms both inputs into the same size of the output with the correct order

of values in the vectors. In fact, for any dimension and size of the input, we can always use the Einstein summation functions followed by reshape functions to transform the input data to the correct size we need. In our method, we insert the *einsum* operation nodes for any connections between the sub-graphs we partitioned. Each *einsum* operation node has an Einstein summation function followed by a reshape function with the right arguments to ensure the correct data flow between the sub-graphs. The outline for the AMTC method is shown in Alg.1.

Algorithm 1. Accelerated Model evaluations on Tensor grids using Computational graph transformations (AMTC) algorithm

1: Specify a computational graph for a computational model
2: Run Kahn's algorithm [50] to determine the correct order for the operations to be evaluated
3: Generate the operations' dependency information and store it as an influence matrix
4: Group the operations that share the same dependency information and form the sub-graphs
5: Insert the *einsum* operation nodes with the right argument between the sub-graphs
6: Evaluate the modified computational graph with the correct input points

To demonstrate how our algorithm modifies the computational graph and reduces the evaluation costs, we show a comparison of computational graphs with and without using AMTC in Fig. 3.2. Assuming we evaluate the function in (3.3) at full-grid quadrature points of the uncertain inputs, this figure shows the computational graphs with and without using the AMTC method. The size of the data flow is also labeled in the graphs and the partitioned sub-graphs are indicated on the modified computational graph. In this case, without using the AMTC method, both inputs u_1 and u_2 have the size of $k^2 \times 1$, and thus each operation in the computational graph is evaluated for k^2 number of times. However, with the AMTC method, the computational graph is partitioned into three sub-graphs. The operations in sub-graph 1 are evaluated on the k quadrature points in u_1 dimension while operations in sub-graph 2 are evaluated on the k quadrature points in u_2 dimension. Two *einsum* operations are inserted connecting sub-graph 1 and sub-graph 2 to sub-graph 3, so that the outputs of the first two sub-graphs are modified into the correct size and sub-graph 3 can be evaluated k^2 times to gather the same output values as the current method.



(a) Computational graph without using AMTC

(b) Computational graph using AMTC

Figure 3.2. Computational graphs with data size for full-grid quadrature points evaluation on $f = cos(u_1) + exp(-u_2)$

3.3.2 Software implementation

The AMTC method can only be implemented from the data access layer of a software package where we have access to the computational graph and data structure of a computer program. This is possible thanks to the Computational System Design Language (CSDL) package[1] [26]. CSDL is an embedded domain-specific language targeting large-scale multidisciplinary design analysis and optimization problems. With CSDL, the user defines the model as a sequence of operations by using a software interface that is highly expressive and natural (CSDL code resembles regular Python).

We show a demonstration graph for the implementation of AMTC on CSDL in Fig. 3.3. The CSDL package has a unique three-stage compiler, which includes front-end, middle-end, and back-end. Using the front-end, the user defines the model and specifies the problem that needs to be solved. Next, the middle-end constructs a computational graph for that model, and the AMTC acts as a graph transformation method to generate the modified computational graph that is more efficient to evaluate. Finally, at the back-end, it creates an executable script based on the modified computational graph using an automatic code generation approach.

Below is a coarse outline of how CSDL and our proposed method interact:

1. User defines the model and identifies the uncertain inputs.
2. Generate the quadrature points according to the distributions of the input variables.
3. Generate the computational graph from the numerical model the user defines.
4. Modify the computational graph as in Alg. 1.
5. Generate an executable back-end.
6. Post-process the output data to calculate the desired quantities of the output.

[1] CSDL software repository: https://lsdolab.github.io/csdl/

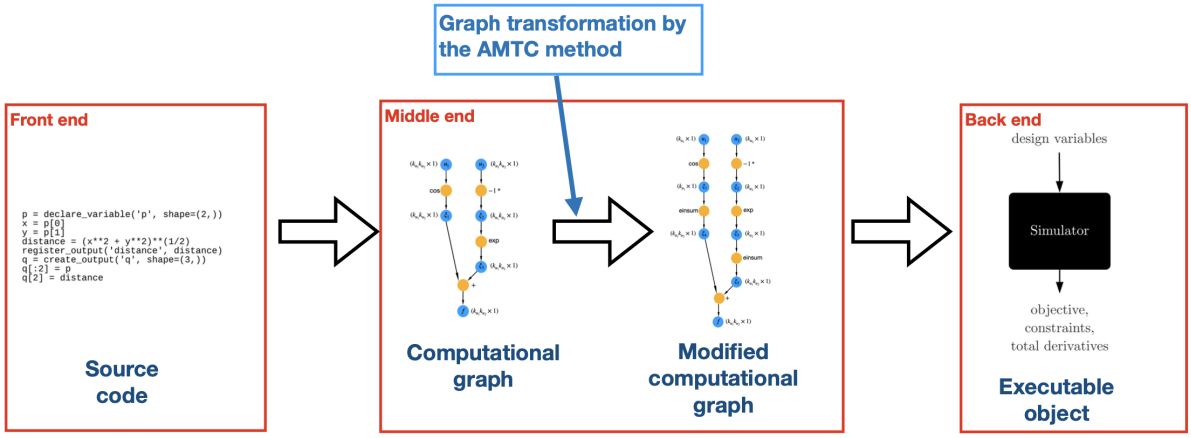


Figure 3.3. Demonstration for the implementation of AMTC on CSDL

3.4 Numerical Results

We investigate the performance of AMTC by using it with the full-grid NIPC method on three low-dimensional UQ problems involving different computational models. The AMTC method has been implemented in the middle-end of CSDL compiler as a graph transformation method, and all of the computational models involved in the test problems are built in CSDL in order to use AMTC to accelerate its tensor-grid evaluations.

3.4.1 3-dimensional UQ problem with an analytical piston simulation model

We first consider a UQ problem involving an analytical non-linear model of the cycle time of a piston. This problem is adapted from [9]. The piston cycle time C in seconds is expressed as

$$C = 2\pi\sqrt{\frac{M}{k + S^2\frac{P_0V_0T_a}{T_0V^2}}}, \tag{3.6}$$

with

$$V = \frac{S}{2k}\left(\sqrt{A^2 + 4k\frac{P_0V_0}{T_0}T_a} - A\right) \text{ and } A = P_0S + 19.62M - \frac{kV_0}{S}. \tag{3.7}$$

In this problem, three of the input parameters are normal random variables, and the rest of them are deterministic variables. The values and descriptions for all of the variables are shown in Table 3.2. The objective of this problem is to compute the expectation value of the piston cycle time, namely, $E[C]$.

Table 3.2. Input parameters and ranges for the piston problem

Input parameters	Range	Description
M	$N(50, 10)$	Piston weight (kg)
S	$N(0.01, 0.002)$	Piston surface area (m^2)
V_0	$N(0.005, 0.001)$	Initial gas volume (m^3)
k	3000	Spring coefficient (N/m)
P_0	100000	Atmospheric pressure (N/m^2)
T_a	293	Ambient temperature (K)
T_0	350	Filling gas temperature (K)

This UQ problem and those that follow are all solved using five methods: integration-based NIPC method using the full-grid quadrature points (full-grid NIPC), full-grid NIPC with the AMTC method (full-grid NIPC with AMTC); integration-based NIPC method using the designed quadrature method (designed quadrature NIPC); kriging and the Monte Carlo method. The kriging method is implemented in its basic form without using adaptive sampling and hyperparameter tuning. The sample points are generated by random sampling, and the kriging surrogate model is trained using the surrogate modelling toolbox in [12].

In our numerical experiments, the full-grid NIPC and the full-grid NIPC with AMTC methods always generate the same results. Fig. 3.4a shows the model evaluation time in terms of the number of equivalent model evaluations for the full-grid NIPC method with and without AMTC. The results show that the AMTC method brought a consistent 50%-60% reduction in evaluation time. In Fig. 3.4b, the convergence plots of these five methods are compared for the UQ result. The percentage errors are calculated with respect to the results of the full-grid NIPC using 400 quadrature points. For this UQ problem, the NIPC methods completely outperform the Monte Carlo and kriging methods. This is because this UQ problem is fairly low-dimensional (only three uncertain inputs), and the function is generally smooth. In this kind of UQ problem,

PCE-related methods are often the most efficient choices. Among the NIPC methods, the designed quadrature NIPC performs better than the full-grid NIPC, as the designed quadrature NIPC requires a smaller number of quadrature points to achieve the same level of accuracy on integration. However, we see a dramatic speed-up after applying the AMTC method to the full-grid NIPC; the total model evaluation time is reduced by almost an order of magnitude, making the full-grid NIPC with AMTC the most efficient method among other methods.

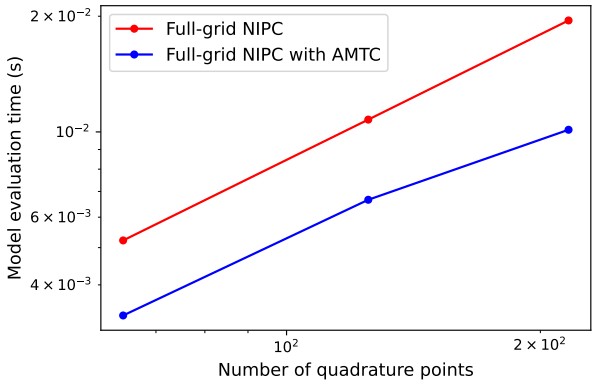
The major speed-up that AMTC brought for full-grid NIPC can be explained by the sparsity of the influence matrix that describes the dependency information of the computational model in this UQ problem. We show the number of dependent operations for each uncertain input in this computational model in Tab. 3.3. From this table, we observe that out of the 65 operations in the computational graph, 50% of the operations depend on the uncertain input M and 60% of the operations depend on uncertain inputs S and V_0. This means, a large portion of the operations are not dependent on all of the uncertain inputs and a significant number of repeated evaluations are eliminated from the operational level with the AMTC method. As this computational model only comprises basic arithmetic operations, the repeated evaluations AMTC eliminated significantly reduces the overall model evaluation time, making the full-grid NIPC the most efficient UQ method.

Table 3.3. Number of dependent operations for each uncertain input in the piston model

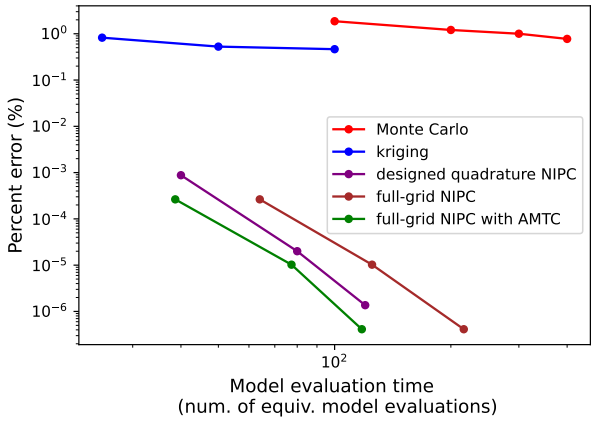
Uncertain input	No. of dependent operations	Total operations
M	31	65
S	43	65
V_0	45	65

3.4.2 3-dimensional UQ problem with a UAV design multidisciplinary model

The second UQ problem we consider involves a low-fidelity multidisciplinary model that computes the total energy stored for a laser-beam-powered unmanned aerial vehicle (UAV) cruising around a ground station for one cycle while being charged by the laser beam. The UQ



(a) Comparison of the full-grid NIPC methods with and without applying the AMTC method



(b) Convergence of the UQ results for the five methods

Figure 3.4. UQ results on the 3D piston problem

problem aims to compute the expectation of the total energy stored under three uncertain inputs. We show the mission plot in Fig. 3.5 and the input parameters in Tab. 3.4. The computational model involves five disciplines, comprising beam propagation, weights, aerodynamics, and performance models. Further details can be found in [97].

The performance comparison of the full-grid NIPC method with and without AMTC is shown in Fig. 3.6a, and the convergence results for all of the UQ methods are shown in Fig. 3.6b. We observe that the AMTC method provided approximately a 90% speed-up in the model evaluation time for the full-grid NIPC method. As a result, full-grid NIPC with AMTC is more efficient to use than all of the other UQ methods we implemented. This is not surprising to see, as in the multidisciplinary model, the parameter uncertainties typically come from different disciplines and there may exist some uncertain inputs that only affect a small portion of the operations. In this scenario, the acceleration provided by the AMTC method can be extremely significant when evaluating on the full-grid quadrature points. From the dependency information shown in Tab.3.5, the uncertain input η affects only less than 10% of the operations in the computational model, while the other two operations affect roughly 50% of the operations.

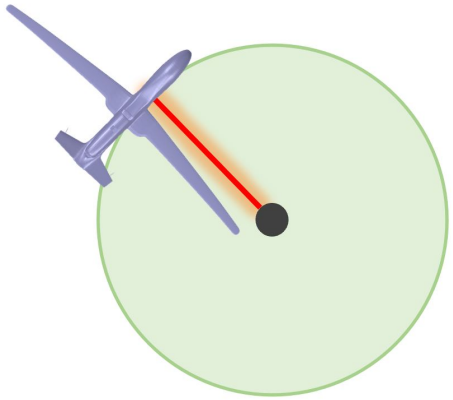
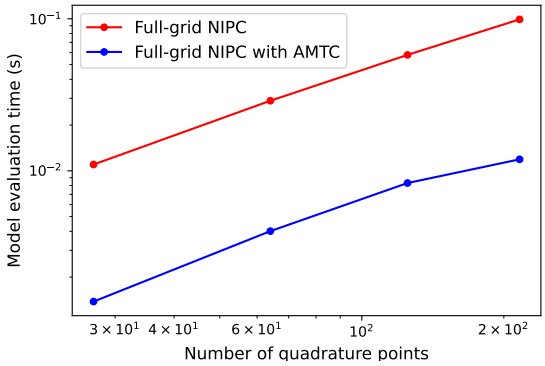
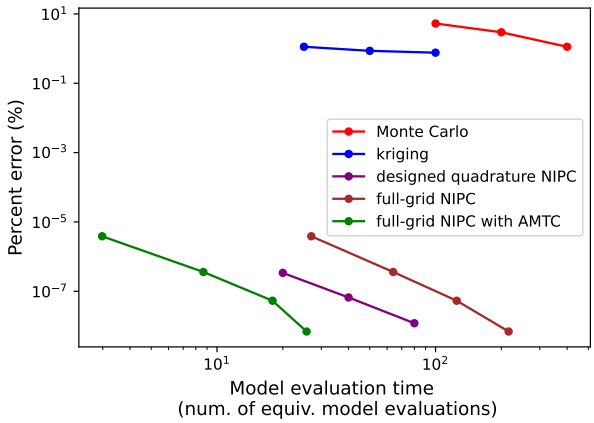


Figure 3.5. A circular cruise mission around a ground station



(a) Comparison of the full-grid NIPC methods with and without applying the AMTC method



(b) Convergence of the UQ results for the five methods

Figure 3.6. UQ results on the 3D multi-disciplinary problem

Table 3.4. Input parameters and ranges for the multidisciplinary model

Input parameters	Range	Description
V	$N(100, 20)$	Velocity (m/s)
h	$N(10,000, 2,000)$	Altitude(m)
η	$N(0.2, 0.03)$	Atmospheric extinction

Table 3.5. Number of dependent operations for each uncertain input in the multidisciplinary model

Uncertain input	No. of dependent operations	Total operations
V	162	285
h	141	285
η	21	285

3.4.3 2-dimensional UQ problem with a eVTOL design multi-point model

The third UQ problem we consider is a multi-point mission analysis problem involving an electric vertical takeoff and landing (eVTOL) aircraft. The problem is adapted from [89]. The UQ problem aims to compute the expectation of the total energy consumption of a lift-plus-cruise eVTOL aircraft concept (Fig.3.7) in a two-segment mission including climb and cruise. Two uncertain inputs are considered in this problem, which are the flight speeds at two mission segments. The details of the properties for the climb and cruise segment are presented in Table 3.6. For each flight segment, we perform a single-point analysis halfway through the stage and use the vortex-lattice method (VLM), an incompressible and inviscid aerodynamic analysis method to compute the lift and drag forces. The details of the computational model can be found in [103].

Table 3.6. Properties for climb and cruise segments of the flight mission

	Climb segment	Cruise segment
Initial altitude (ft)	6,000	10,000
Final altitude (ft)	10,000	10,000
Flight path angle γ (deg)	20	0
Range R (nmi)	R_{climb}	$37.5 - R_{\text{climb}}$
Speed V (Mach)	$N(0.3, 0.03)$	$N(0.5, 0.05)$

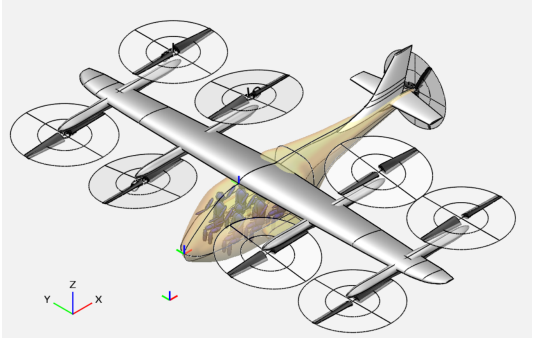
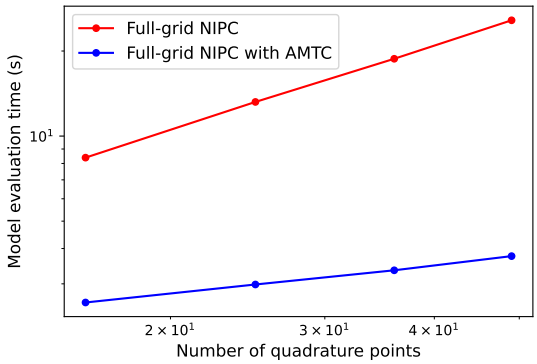


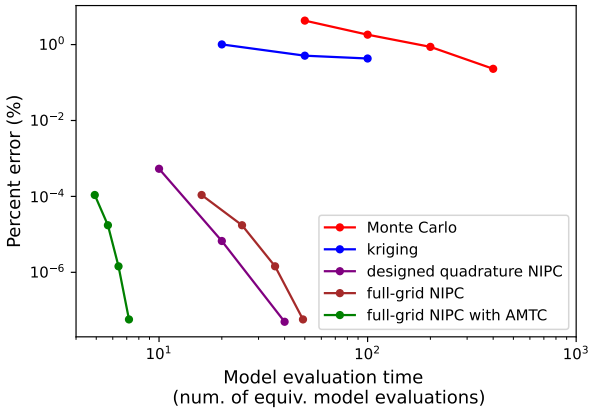
Figure 3.7. Representation of the eVTOL aircraft [89], credit to NASA

We show the performance comparison of the full-grid NIPC method with and without AMTC in Fig. 3.8a and the UQ convergence plot comparing the five UQ methods in Fig. 3.8b. For this problem, the AMTC provided a 70-90% percent reduction in evaluation time and the reduction got more significant as we increase the number of quadrature points. As a result, the full-grid NIPC with AMTC method is significantly more efficient to use than the other UQ methods. The dependency information in Tab. 3.7 shows roughly 40% of the operations depend on each uncertain input. The sparsity of the influence matrix is present because, for this multi-point problem, we have two uncertain inputs each only affecting the aerodynamic analysis at one point. Although the output of the model is based on the results from both aerodynamic analyses, each aerodynamic analysis is only evaluated on the quadrature points in one-dimensional input space by using the AMTC method. This dramatically accelerated its model evaluation time on full-grid quadrature points. In fact, the performance of AMTC can be more significant for many multi-point problems involving multi-point analyses and multiple uncertain inputs.

We recognize that on this test problem, given the straightforward computational graph structure, our method's performance could be replicated by manually conducting two distinct sets of evaluations with the VLM solver and subsequently integrating these evaluations using tensor operations. However, in a practical design workflow, where UQ problems might undergo frequent reformulations, our method can automatically achieve this reduction in computational cost, sparing users the effort of manual implementation.



(a) Comparison of the full-grid NIPC methods with and without applying the AMTC method



(b) Convergence of the UQ results for the five methods

Figure 3.8. UQ results on the 2D multi-point problem

Table 3.7. Number of dependent operations for each uncertain input in the multi-point model

Uncertain input	No. of dependent operations	Total operations
V_{climb}	605	1505
V_{cruise}	604	1505

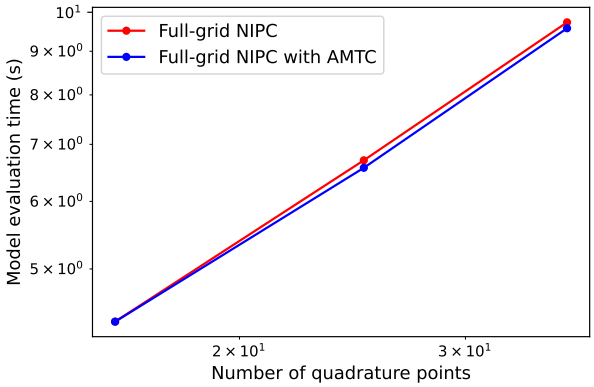
3.4.4 2-dimensional UQ problem with a rotor aerodynamic analysis model

The fourth UQ problem we consider is a rotor aerodynamic analysis involving a blade element momentum model. The description of the model can be found in [81]. In this UQ problem, we aim to compute the expectation of the total torque generated by the rotor, given two uncertain inputs, rotational rotor speed, and axial free-stream velocity. The description of the uncertain inputs is shown in Tab. 3.8.

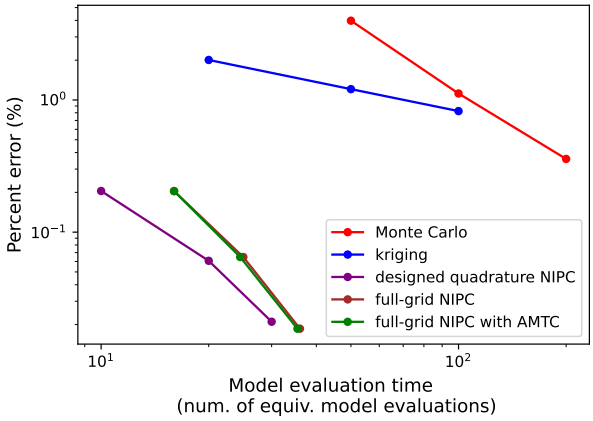
The performance comparison of the full-grid NIPC method with and without AMTC is shown in Fig. 3.9a. The convergence results for all of the UQ methods are shown in Fig. 3.9b, and the dependency information is shown in Tab. 3.9. In Tab. 3.9, we observe that the influence matrix here is also sparse, as roughly 55% of the operations in the computational model are dependent on each of the uncertain inputs. However, we observe little acceleration for the full-grid NIPC using AMTC in Fig. 3.9a. As a result, the full-grid NIPC is outperformed by the designed quadrature method for the UQ results in Fig. 3.9b. This is because the dependency information is not the only factor that affects the AMTC method's performance. The other factor is each operation's evaluation time. In this model, there exists an implicit operation which also counts as one operation node but its evaluation time takes more than 95% of the model evaluation time. Since the AMTC method does not reduce the number of model evaluations on this implicit operation, the reduction in model evaluation time by AMTC becomes insignificant.

Table 3.8. Input parameters and ranges for the rotor model

Input parameters	Range	Description
Ω	$N(100,10)$	Rotational rotor speed (rad/s)
V_x	$N(50,5)$	Axial free-stream velocity(m/s)



(a) Comparison of the full-grid NIPC methods with and without applying the AMTC method



(b) Convergence of the UQ results for the five methods

Figure 3.9. UQ results on the 2D rotor problem

Table 3.9. Number of dependent operations for each uncertain input in the rotor model

Uncertain input	No. of dependent operations	Total operations
Ω	398	708
V_x	423	708

Acknowledgments

The material presented in this chapter is, in part, based upon work supported by NASA under award No. 80NSSC21M0070 and DARPA under grant No. D23AP00028-00.

Chapter 3, in part, is a reprint of the material [104] as it appears in Wang, B., Sperry, M., Gandarillas, V. E., and Hwang, J. T., "Accelerating model evaluations in uncertainty propagation on tensor grids using computational graph transformations," Aerospace Science and Technology, vol. 145, 2024, pp. 108843. The dissertation author was the primary investigator and author of this chapter.

Chapter 4

Graph-accelerated non-intrusive polynomial chaos (NIPC) using partially tensor-structured quadrature rules

Chapter 3 introduces a new method called *Accelerated Model evaluations on Tensor grids using Computational graph transformations* (AMTC) [104]. This method reduces the computational cost of model evaluations on tensor-grid inputs by modifying the computational graph of the model within the middle end of a modeling language's three-stage compiler. This modification eliminates redundant evaluations at the operation level caused by the tensor structure of the inputs. The AMTC method has been applied with integration-based NIPC to solve UQ problems, under the assumption of a specific type of sparsity within the computational graphs of the computational models. This integration, termed the *graph-accelerated NIPC* method, has demonstrated superior efficiency in various UQ problems involving multidisciplinary systems. However, the graph-accelerated NIPC method has so far only been implemented with a full-grid Gauss quadrature rule, targeting low-dimensional UQ problems.

This chapter introduces a novel framework aimed at producing a quadrature rule possessing a specified tensor structure, tailored for use with the graph-accelerated NIPC method in addressing the UQ problem. Within this framework, we first identify a suitable tensor structure option for the quadrature rule through an analysis of the computational model. Subsequently, we employ the designed quadrature method to construct the quadrature rule that adheres to this

tensor structure. The resulting quadrature rule is strategically designed to leverage the sparsity inherent in the computational graph of the model, thus maximizing efficiency when employing the AMTC method for accelerating tensor-grid evaluations.

This method has been applied to two UQ problems, one with four uncertain inputs and one with six uncertain inputs, both derived from practical aircraft design problems involving multidisciplinary systems. The proposed method produces new partially tensor-structured quadrature rules distinct from existing ones for both test problems. These partially structured quadrature rules also outperform existing methods when employed with the graph-accelerated NIPC method for solving UQ problems.

4.1 Numerical quadrature rules

This chapter focuses on employing the integration-based NIPC method for solving multi-dimensional UQ problems. In this context, the integration-based NIPC method is essentially solving the integration problem as in (2.15). We denote the integration problem as

$$I(f) = \int_{\Gamma} f(u)\rho(u)du,$$

where $u = (u_1, \ldots, u_d) \in R^d$. Solving this integration problem often involves utilizing the numerical quadrature approach, which approximates the integral as a weighted sum of function evaluations at specific quadrature points:

$$I(f) \approx \sum_{i=1}^{n} w^{(i)} f(u^{(i)}), \tag{4.1}$$

where $u^{(i)} \in \Gamma$ and $w^{(i)} > 0$ represent the nodes and weights, respectively, to be determined by the quadrature rule.

The objective of the quadrature rule is to effectively approximate a large set of functions with a minimal number of function evaluations. This is typically achieved by ensuring equality

conditions hold for all functions f within a polynomial subspace Π, as expressed by:

$$\int_{\Gamma} f(u)\rho(u)du = \sum_{i=1}^{n} w^{(i)} f(u^{(i)}) \quad \text{for all} \quad f \in \Pi_r. \tag{4.2}$$

Here, we consider the cross-polynomial subspace with a total polynomial order of r, defined as:

$$\Pi_r = \text{span}\{u^{\alpha} \mid \alpha \in \Lambda_r\}, \tag{4.3}$$

where

$$\alpha = (\alpha_1, \ldots, \alpha_d), \quad u^{\alpha} = \prod_{j=1}^{d} (u_j)^{\alpha_j} \tag{4.4}$$

and

$$\Lambda_r = \left\{ \alpha \mid \sum_{i=1}^{d} \alpha_i \leq r \right\}. \tag{4.5}$$

In many numerous integration problems, the evaluation functions exhibit smooth behavior and can be accurately approximated by the polynomials within the cross-polynomial subspace. In such instances, the quadrature rule enforcing Equation (4.2) can prove to be highly effective.

Gauss quadrature rule

One approach to generating the quadrature points is through the use of the Gauss quadrature rule [31]. In the one-dimensional case, the quadrature points $u^{(1)}, \ldots, u^{(k)}$ are the roots of the orthogonal polynomial $p_k(u)$ with degree k. The polynomials here are defined in the same way as the PCE basis functions, and they also satisfy orthogonality:

$$\langle p_i, p_j \rangle = \int_{\Gamma} p_i(u) p_j(u) \rho(u) du = \delta_{ij}. \tag{4.6}$$

The weights of the Gauss quadrature rule, $w^{(1)}, \ldots, w^{(k)}$ can be computed by solving the moment-matching equations:

$$\sum_{i=1}^{k} p_j\left(u^{(i)}\right) w^{(i)} = \begin{cases} 1 & \text{if } j = 0 \\ 0 & \text{for } j = 1, \ldots, k-1. \end{cases} \tag{4.7}$$

In this approach, the univariate Gauss quadrature rule with k nodes and weights can exactly integrate the univariate polynomials up to degree $2k-1$.

For multi-dimensional integrations, the Gauss quadrature rule extends the univariate rule to form a tensor structure, Assuming k quadrature points in each dimension, the integral is approximated as

$$I \approx \sum_{i_1=0}^{k} \ldots \sum_{i_d=0}^{k} w_1^{(i_1)} \ldots w_d^{(i_d)} f(u_1^{(i_1)}, \ldots, u_d^{(i_d)}) \rho(u_1^{(i_1)}, \ldots, u_n^{(i_d)}), \tag{4.8}$$

where $(u_i^{(1)}, \ldots, u_i^{(k)})$ and $(w_i^{(1)}, \ldots, w_i^{(k)})$ are the nodes and weights for the 1D Gauss quadrature rule in the u_i dimension, respectively. We represent the full-grid quadrature points as

$$u = u_1^k \times \cdots \times u_d^k, \tag{4.9}$$

where $u_i^k := \{u_i^{(j)}\}_{j=1}^k$. Indeed, while this method allows exact integration of cross-polynomials up to a total order of $2k-1$, it suffers from the curse of dimensionality. This is because the total number of quadrature points increases exponentially with the number of dimensions, as $n = k^d$.

To mitigate the curse of dimensionality the Gauss quadrature rule suffers from, the Smolyak sparse grid method [90] offers a solution. This method strategically drops higher-order cross terms in the full-grid Gauss quadrature points, reducing the number of required quadrature points while preserving the accuracy of the quadrature rule to a significant extent. The sparse

grid quadrature points can be expressed as:

$$u = \bigcup_{\ell-d+1 \le |\mathbf{i}| \le \ell} \left(u_1^{i_1} \times \cdots \times u_d^{i_n} \right), \tag{4.10}$$

where ℓ is the level of construction.

Designed quadrature method

For high-dimensional integration problems, a more effective quadrature rule is offered by the designed quadrature method proposed by Keshavarzzadeh et al. [55]. The core idea of the designed quadrature method is to numerically generate a quadrature rule that satisfies the multi-variate moment-matching equations:

$$\sum_{i=1}^{n} \Phi_{i'} \left(u^{(i)} \right) w^{(i)} = \begin{cases} 1 & \text{if } |i'| = 0 \\ 0 & \text{for } \alpha \in 0 < |i'| < k, \end{cases} \tag{4.11}$$

where

$$\pi_\alpha = \prod_{i=1}^{d} p_{\alpha_i}^{(i)}(u_i) \tag{4.12}$$

and $p_j^{(i)}$ is the j-th orthogonal polynomial in the u_i dimension. If the nodes and weights of the quadrature rule satisfy the equations in (5.24), then the quadrature rules can exactly integrate cross-term polynomials up to $(2k-1)$th order. We denote the quadrature points and the weights as $u := (u^{(1)}, \ldots, u^{(n)})$ and the weights $w := (w^{(1)}, \ldots, w^{(n)})$ respectively. Then, the moment-matching equations in (5.24) can be written as residual equations:

$$\mathscr{R}(u, w) = 0, \tag{4.13}$$

where $\mathscr{R} : R^{d \times n} \times R^n \to R^n$.

The designed quadrature method approximates the solution of the moment-matching

equations by solving an optimization problem formulated as

$$\begin{aligned} &\min_{u,w} \|\mathscr{R}(u,w)\|_2 \\ &\text{subject to } u^{(j)} \in \Gamma, \quad j = 1,\ldots,n, \\ &\qquad w^{(j)} > 0, \quad j = 1,\ldots,n. \end{aligned} \tag{4.14}$$

The objective of this optimization, as shown in Equation (4.14), is to minimize the L^2 norm of the residual equations while constraining the nodes to lie within the support range and ensuring that the weights are positive. This method has demonstrated superior effectiveness compared to the sparse-grid Gauss quadrature rule for many high-dimensional integration problems. The number of quadrature points required is typically chosen as $n = \eta \frac{(2d)^{k-1}}{(k-1)!}$, where $\eta \in [0.5, 0.9]$ serves as a tuning parameter.

4.2 Methodology

This chapter proposes a new framework that generates a tensor-structured quadrature rule that is specifically tailored for a given computational model. This quadrature rule is intended to be used with the graph-accelerated NIPC method to efficiently solve the UQ problem.

4.2.1 Generation of tensor-structured quadrature rules

We use the degree of polynomials that a quadrature rule can exactly integrate as the measure of its accuracy. Under this criterion, if we maintain a non-tensorial structure for the quadrature rule, the most effective strategy is to utilize the designed quadrature method. In this scenario, the quadrature points can be expressed as:

$$u = u^n_{\{1,\ldots,d\}}. \tag{4.15}$$

The nodes and weights in the quadrature rule are determined by solving the optimization problem outlined in (4.14), and the number of quadrature points is chosen as

$$n = \eta \frac{(2d)^{k-1}}{(k-1)!}, \quad \eta \in [0.5, 0.9]. \tag{4.16}$$

Adopting this method ensures that the resulting quadrature rule can effectively integrate polynomials up to $(2k-1)$th order. If a fully tensorial structure for the quadrature points is necessary, the optimal method to employ is the full-grid Gauss quadrature method. In this approach, the quadrature points are expressed as:

$$u = u^k_{\{1\}} \times \cdots \times u^k_{\{d\}}, \tag{4.17}$$

where $u^k_{\{i\}}$ represents the 1D Gauss quadrature points in u_i dimension. The sets of 1D Gauss quadrature points in each dimension are determined separately to ensure the nodes and weights can exactly integrate the univariate polynomials up to $(2k-1)$th order. Maintaining a fully tensorial structure of the 1D Gauss quadrature points and weights ensures that the full-grid quadrature rule can exactly integrate cross-polynomials up to the $(2k-1)$th order, albeit while requiring $n = k^d$ quadrature points. In fact, in each dimension, the 1D Gauss quadrature points and weights correspond to the global minimum of the optimization problem outlined in (4.14). Assuming that the optimization problem in (4.14) always finds the global minimum, employing the full-grid Gauss quadrature rule is equivalent to applying the designed quadrature method in each dimension and subsequently forming a fully tensorial structure between the 1D solutions.

In addition to the fully tensorial and non-tensorial structure options, we can also preserve a partially tensorial structure for the quadrature points while ensuring accurate integration of all polynomials up to a specified order. For instance, in a 4D integration problem, we could

maintain a partially tensorial structure as:

$$u = u^{n_1}_{\{1,2\}} \times u^{n_2}_{\{3,4\}}, \tag{4.18}$$

where a tensorial structure is retained between the quadrature points in the space of u_1, u_2 and the quadrature points in the space of u_3, u_4. If the quadrature rules in each space are selected such that the quadrature points can precisely integrate polynomials up to the same order, then this partially tensorial quadrature rule can also precisely integrate the polynomials to the same order in the space of u. The total number of tensorial structure options for the quadrature rule can be represented by the Bell number from combinatorial mathematics, which is given by:

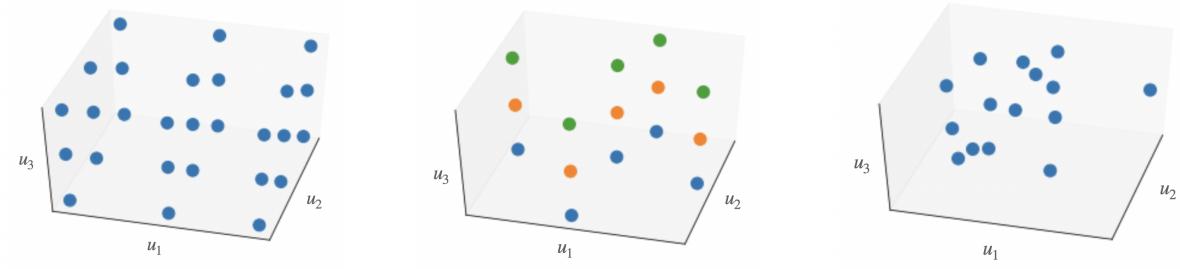
$$B_d = \sum_{i=0}^{d} \left\{ \begin{array}{c} i \\ d \end{array} \right\}. \tag{4.19}$$

For example, for $d = 3$, $B_3 = 5$, the five options of the tensorial structure are:

$$\begin{aligned} u &= u^{n}_{\{1,2,3\}}, \\ u &= u^{n_1}_{\{1,2\}} \times u^{n_2}_{\{3\}}, \\ u &= u^{n_1}_{\{1,3\}} \times u^{n_2}_{\{2\}}, \\ u &= u^{n_1}_{\{2,3\}} \times u^{n_2}_{\{1\}}, \\ u &= u^{n_1}_{\{1\}} \times u^{n_2}_{\{2\}} \times u^{n_3}_{\{3\}}. \end{aligned} \tag{4.20}$$

Visualizations of quadrature points with different tensorial structures can be found in Fig. 4.1.

Given a specific tensorial structure option for the quadrature rule, we need to decide on the method for generating the quadrature points within each space and determine the requisite number of quadrature points to attain the desired level of accuracy. In our method, for a particular tensorial structure of the quadrature rule, we employ the designed quadrature method in each space. This method determines the nodes and weights to accurately integrate polynomials in



(a) $u = u^n_{\{1\}} \times u^n_{\{2\}} \times u^n_{\{3\}}$ **(b)** $u = u^{n_1}_{\{1,2\}} \times u^{n_2}_{\{3\}}$ **(c)** $u = u^n_{\{1,2,3\}}$

Figure 4.1. Visualizations of quadrature points with different tensorial structure options

that space up to a specified order. Adapted from the Gauss quadrature rule, we use the level of accuracy k for a quadrature rule to indicate its capability of precisely integrating polynomials up to the $(2k-1)$th order in the integration space. The number of quadrature points utilized in the designed quadrature method is given by (4.16), tailored for high-dimensional integration problems. In our setting, we also utilize the designed quadrature method for relatively low-dimensional spaces. Therefore, we choose the required number of quadrature points using:

$$n = \begin{cases} \frac{k^d}{d} & \text{for } d \leq 2 \\ 0.9\frac{(2d)^{k-1}}{(k-1)!} & \text{for } d > 2. \end{cases} \tag{4.21}$$

When the number of dimensions is greater than two, we follow the designed quadrature rule in (4.16) with $\eta = 0.9$. However, for 1D and 2D cases, we choose $n = \frac{k^d}{d}$. In this way, in the 1D space, it chooses k quadrature points to achieve k level of accuracy, which matches the Gauss quadrature rule. Consequently, for the 1D space, we can avoid solving the optimization problem in the designed quadrature method and directly use the 1D Gauss quadrature rule, as it represents the exact solution of the designed quadrature method. This approach results in generating the full-grid Gauss quadrature rule if a fully tensorial structure option is chosen, and generating the designed quadrature rule if a non-tensorial structure option is chosen.

4.2.2 Graph-accelerated tensor-grid evaluations

When not using the AMTC method for model evaluations, tensor-structured quadrature rules often underperform compared to non-tensorial quadrature rules. This is because tensor-structured quadrature rules always require evaluations of the model at more input points than non-tensorial quadrature rules to achieve the same level of accuracy. However, when utilizing the computational graph transformation method, AMTC, redundant operations incurred by the tensorial structure of the input points are eliminated at the operation level. Consequently, the overall cost of evaluating the model is no longer directly tied to the number of quadrature points utilized. This gives us the reason for devising a custom tensor-structured quadrature rule tailored for a given computational model.

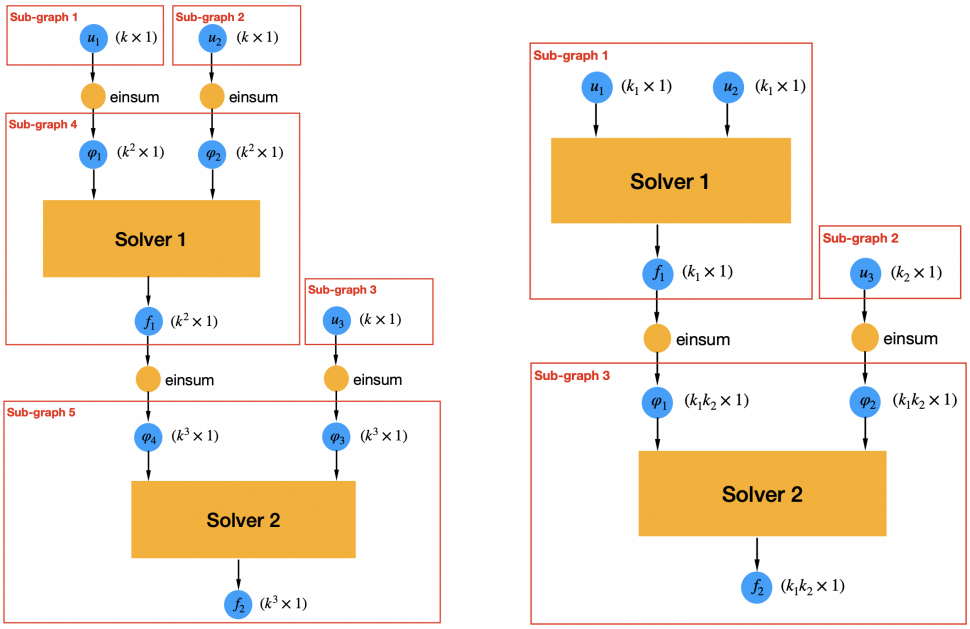
For instance, consider a 3D UQ problem involving two computationally expensive solvers. The computational model can be written as

$$\begin{aligned} f_1 &= \mathscr{F}_1(u_1, u_2), \\ f_2 &= \mathscr{F}_2(f_1, u_3), \end{aligned} \tag{4.22}$$

where $\mathscr{F}_1$ and $\mathscr{F}_2$ represent the first and second solver, respectively, and the UQ problem aims to compute the risk measures of f_2 under the effect of the three uncertain inputs. In this case, we have two uncertain inputs associated with solver 1, and one uncertain input associated with solver 2. Consequently, the operations in solver 1 are dependent on u_1 and u_2 while the operations in solver 2 are dependent on u_1, u_2, and u_3. In such scenario, an optimal quadrature rule can be easily formulated following a partially tensorial structure:

$$u = u_{\{1,2\}}^{k_1} \times u_{\{3\}}^{k_2}. \tag{4.23}$$

The corresponding model evaluations can be significantly accelerated using the AMTC method. To demonstrate, we show the computational graphs when performing model evaluations on



(a) $u = u^k_{\{1\}} \times u^k_{\{2\}} \times u^k_{\{3\}}$ **(b)** $u = u^{k_1}_{\{1,2\}} \times u^{k_2}_{\{3\}}$

Figure 4.2. Computational graphs after AMTC on fully and partially tensor structured input points

fully tensorial quadrature points $u^k_{\{1\}} \times u^k_{\{2\}} \times u^k_{\{3\}}$ and partially tensorial quadrature points $u^{k_1}_{\{1,2\}} \times u^{k_2}_{\{3\}}$ after applying AMTC in Fig. 4.2. We observe that, in both cases, AMTC generates modified computational graphs that reduce the number of evaluations on solver 1. In comparison, the partially tensor-structured quadrature rule is more favorable than the fully tensor-structured quadrature rule, as both the u_1 and u_2 affect the same operations on the computational graph, so there is no need to maintain a tensor structure between u_1 and u_2 like in the fully tensor-structured quadrature rule. For the partially tensor-structured quadrature rule, by grouping the space of u_1 and u_2, the total number of quadrature points in this space, k_1, is smaller than the full-grid quadrature rule which has k^2 quadrature points in this space. Consequently, utilizing the partially tensor-structured quadrature rule results in fewer evaluations for both solver 1 and solver 2.

4.2.3 Finding a desirable tensor structure option

For specific UQ problems, particularly those involving multidisciplinary systems, certain partially tensor-structured quadrature rules demonstrate superior performance compared to

Table 4.1. Operations dependency information for function $f = cos(u_1) + exp(-u_2)$

$D(\varphi_i, u_j)$	u_1	u_2
φ_1	1	0
φ_2	0	1
φ_3	0	1
φ_4	1	1

existing ones when utilized with the graph-accelerated NIPC method. However, identifying an optimal tensor-structure option may be challenging.

Theoretically, we can estimate the model evaluation costs after applying AMTC for each tensor-structured rule. To do that, for a computational model $f = \mathscr{F}(u)$, we denote the data and operations nodes as ξ_i and φ_j, respectively. Here l represents the number of operations in the computational graph. For example, consider the function

$$f = cos(u_1) + exp(-u_2), \tag{4.24}$$

with its corresponding computational graph described as

$$\begin{aligned} \xi_1 &= \varphi_1(u_1) = cos(u_1); \\ \xi_2 &= \varphi_2(u_2) = -u_2; \\ \xi_3 &= \varphi_3(\xi_1) = exp(\xi_2); \\ f &= \varphi_4(\xi_2, \xi_3) = \xi_2 + \xi_3. \end{aligned} \tag{4.25}$$

From the computational graph, we can generate the dependency information which stores whether the output of one operation is dependent on one input as a binary number, written as $D(\varphi_i, u_j)$. The dependency information can be viewed in a matrix. The dependency matrix for the function (4.24) is presented in Tab. 4.1, which indicates the dependencies between operations and input variables. To estimate the evaluation cost of each tensorial structure for the quadrature rule to achieve the k-level of accuracy, we also need to store the information regarding the evaluation

cost of each operation on one quadrature point. This information can be represented either by the total number of floating-point operations (FLOPs) or the evaluation time. For the non-tensorial structure, if we assume all operations are dependent on at least one input, no repeated evaluations can be reduced by the AMTC method. Assume that the operations' cost on all of the quadrature points is roughly the same, the total model evaluation cost can be approximated as:

$$\text{Cost} \approx nO\left(f(u)\right) \approx n\sum_{i=1}^{l} O(\varphi_i), \tag{4.26}$$

where $O(\cdot)$ represents the evaluation cost of the function/operation and the number of quadrature points n is chosen as in (5.28). If we maintain a full-grid structure with k quadrature points in each dimension, the evaluation cost without the AMTC method is

$$\text{Cost} \approx k^d O\left(\mathscr{F}(u^{(1)})\right) \approx k^d \sum_{i=1}^{l} O(\varphi_i). \tag{4.27}$$

However, with the AMTC method, each operation in the graph is evaluated only for distinct values. In the full-grid case, if the output of an operation is dependent on $\tilde{d}$ number of uncertain inputs, there are only $k^{\tilde{d}}$ quadrature points in its dependent inputs space, and that operation will be evaluated for $k^{\tilde{d}}$ number of times using the AMTC method. In this case, we omit the cost of the Einstein summation nodes added to the computational graph by the AMTC method, the evaluation cost can be estimated as

$$\text{Cost} \approx \sum_{i=1}^{l} k^{\sum_{j=1}^{d} D(\varphi_i, u_j)} O(\varphi_i). \tag{4.28}$$

Using the same idea, we can also estimate the model evaluations cost for any tensorial structure option of the quadrature points. For example, consider the setting where we have a partially tensorial structure for the quadrature points, such as

$$u = u_{\{1,2\}}^{k_1} \times u_{\{3\}}^{k_2}. \tag{4.29}$$

In our method, k_1 and k_2 are chosen from (5.28) using $d = 1$ and $d = 2$ respectively. In this case, there are k_1 quadrature points in the space of u_1 and u_2, and the k_2 quadrature points in the space of u_3. With the AMTC method, for the operations that are dependent on only u_1, only u_2 or only u_1 and u_2, they will be evaluated n_1 times. Operations dependent on only u_3, will be evaluated n_2 times. Operations not dependent on any input are evaluated once. The rest of the operations are evaluated $n_1 n_2$ times. The model evaluations cost for this tensor-structured quadrature rule can be estimated as

$$\text{Cost} \approx \sum_{i=1}^{l} k_1^{D(\varphi_i,u_1)D(\varphi_i,u_2)} k_2^{D(\varphi_i,u_3)} O(\varphi_i). \tag{4.30}$$

Estimating the total model evaluation cost for all tensor structure options enables us to choose the optimal one with the lowest estimated cost. However, the challenge lies in the fact that the total number of tensor structure options follows the Bell number as shown in (4.19), which scales significantly with the problem dimensions. To avoid this significant overhead cost associated with numerically finding the optimal tensor structure option, we propose an alternative approach. This approach involves selecting a desirable tensor structure option based on our understanding of the computational model and an analysis of the computational graph. By leveraging information from the computational graph and characteristics of the computational model, we can make informed decisions to choose a tensor structure that is likely to lead to efficient model evaluations. This approach balances computational efficiency with accuracy, making it a practical solution for many UQ problems with complicated computational models.

When dealing with computational models involving multiple disciplines or solvers, identifying a desirable tensor-structure option can sometimes be achieved by examining the computational graph at the solver level. For instance, in the example provided in (4.22), we can easily deduce the optimal tensor structure based on insights gained from the computational graph. In other cases, we can perform some analysis on the computational graph to help us decide on the desired tensor-structure option. Our rationale here is based on the observation that complex computational models typically contain one or two computationally expensive operations, such

as linear or non-linear solvers, which significantly contribute to the overall computational cost. We aim to identify uncertain inputs that these expensive operations do not depend on and then establish a tensor structure to minimize the number of evaluations required for these operations, with the utilization of AMTC. To do that, we define the *sparsity ratio* (SR) of an uncertain input as the ratio of its dependent operations' cost over the total model evaluation cost which can be approximated as

$$\mathrm{SR}(u_i) \approx \frac{\sum_{i=1}^{l} O(\varphi_i) D(\varphi_i, u_j)}{O(f(u))}, \tag{4.31}$$

where $\mathrm{SR}(u_i)$ represents the sparsity ratio of u_i and it measures the percentage of the model evaluation cost that is dependent on u_i. We identify the sparse uncertain inputs based on the condition that their sparsity ratio is $< 5\%$. We then partition the uncertain inputs as

$$u = (u_s, u_{ns}), \tag{4.32}$$

where $u_s = (u_{s_1}, \ldots, u_{s_{d_1}}) \in R^{d_1}$ is the set of sparse uncertain inputs and $u_{ns} = (u_{ns_1}, \ldots, u_{ns_{d_2}}) \in R^{d_2}$ is the set of non-sparse uncertain inputs with $d = d_1 + d_2$.

In this case, if all of the uncertain inputs are sparse uncertain inputs, we may want to form a full-grid quadrature rule following the Gauss quadrature method, such that the quadrature points can be written as

$$u = u^k_{\{1\}} \times \ldots \times u^k_{\{d\}}, \tag{4.33}$$

where $u^k_{\{i\}}$ represents the k quadrature points in u_i space. In this way, even though the number of quadrature points increases exponentially as $n = k^d$, the model evaluation cost can be significantly accelerated using the AMTC method which only evaluates each operation on the distinct quadrature points in the space of the uncertain inputs on which it depends. However, in most cases, for most UQ problems, there often only exist a few sparse uncertain inputs. In this case, we suggest choosing the quadrature rule that maintains a tensor structure between the quadrature points in the sparse uncertain input space and quadrature points in the space of

non-sparse uncertain inputs, such that the quadrature points can be written as

$$u = u_{\{ns\}}^{k_1} \times u_{\{s\}}^{k_2}, \tag{4.34}$$

where $u_{\{ns\}}^{k_1}$ represents quadrature points in u_{ns} space with k_1 points and $u_{\{s\}}^{k_2}$ represent the quadrature points in u_s space with k_2 points. This results in a total of $k_1 k_2$ quadrature points in the quadrature rule. However, when using the AMTC method to modify the computational graph, the computational cost on the modified computational graph would scale roughly linearly with k_1 as most of the expensive operations in the computational graph would only depend on u_{ns} and would only be evaluated n_1 times. An alternative option is to formulate the quadrature rule as

$$u = u_{\{ns\}}^{k_1} \times u_{\{s_1\}}^{k} \times \ldots, u_{\{s_{d_1}\}}^{k}, \tag{4.35}$$

which forms a tensor structure between the space of each sparse uncertain input. This option results in more quadrature points but it is easier to generate the quadrature rule and can be more effective to use if we know the sparse uncertain inputs are associated with different solvers.

We show the specific steps of using this method with graph-accelerated NIPC in Alg. 2.

Algorithm 2. Partially tensor-structured quadrature rule for graph-accelerated NIPC

1: Specify a computational graph for a numerical model and the uncertain inputs, U
2: Specify the level of accuracy of the quadrature rule, k
3: Choosing a desired tensor structure option based on the analysis of the computational graph
4: Compute the required number of quadrature points in each space following (4.16)
5: Compute the quadrature points and weights in each space by solving (4.14)
6: Formulate the quadrature rule using the chosen tensorial structure
7: Perform the model evaluations on the tensor-structured quadrature rule using AMTC
8: Post-process the evaluations using NIPC to solve the UQ problem

Table 4.2. Input parameters and ranges for the UAV problem

Uncertain inputs	Distributions
Payload weight W_p (kg)	$\mathcal{N}(90, 10)$
Atmospheric extinction coefficient σ	$\mathcal{N}(0.2, 0.02)$
Flight altitude h (m)	$\mathcal{N}(10{,}000, 1000)$
Flight velocity v (m/s)	$\mathcal{N}(100, 10)$

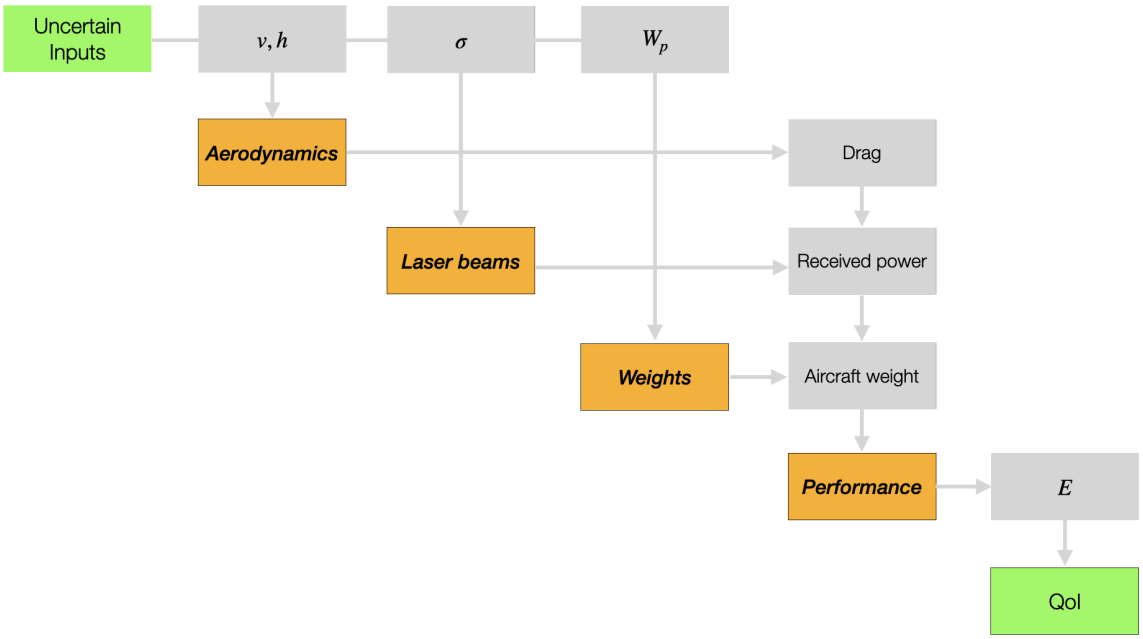


Figure 4.3. Multidisciplinary structure of the UAV model

4.3 Numerical Results

The first test problem is a 4D UQ problem adapted from a UAV design scenario [72]. This computational model assesses the total energy stored during a cruise mission undertaken by a laser-beam-powered UAV as it orbits a ground station for one complete cycle while being recharged by the laser beam emitted from the ground station. The mission is demonstrated in Fig. 3.6. We consider four uncertain inputs associated with the operating conditions, which are flight velocity, flight altitude, payload weight, and atmospheric extinction coefficient. Their distributions are shown in Tab. 6.1. The objective of this UQ problem is to compute the expectation of the stored energy considering the influence of these uncertain inputs.

The computational model used in this problem involves four disciplines: aerodynamics, laser beams, weights, and performance. The multidisciplinary structure of the computational model is shown in Fig. 4.3. For the laser-beam model, we apply the Beers–Lambert law for optical power transfer [56] to compute the power received by the aircraft as

$$P_{\text{rec}} = \eta P_{\text{tra}}, \quad \eta = e^{-\sigma R}, \tag{4.36}$$

where P_{rec} and P_{tra} denote the power received and transmitted, respectively; σ is the atmospheric extinction coefficient which differs across weather conditions; and R is the distance between the aircraft and the power source. In the aerodynamic model, inputs such as flight velocity, flight altitude, and the total aircraft weight are considered. This model is responsible for computing the total drag force exerted on the wing to sustain steady-state flight. Additionally, it determines the lift coefficient and drag coefficient by employing a linear model and a quadratic polar model, respectively, derived from airfoil data. The weights model estimates the weight of each aircraft component using the statistical equations from [79] and outputs the total aircraft weight. The performance model takes inputs such as flight velocity and total drag force to compute the total energy stored during the mission. This computation is based on the following equations:

$$\begin{aligned} n &= \sqrt{\left(\frac{v}{rg}\right)^2 + 1}, \\ P_{\text{req}} &= \frac{D}{n}, \\ E &= (P_{\text{rec}} - P_{\text{req}})t, \end{aligned} \tag{4.37}$$

where n represents the turn load, r is the radius of the mission, g is the gravitational acceleration, D denotes the total drag force, P_{req} is the required power to maintain steady flight, and t represents the mission time.

On this test problem, following our proposed method, sparsity ratios of the uncertain inputs are measured, which are shown in Tab. 4.3. Using the sparsity ratios, two sparse uncertain

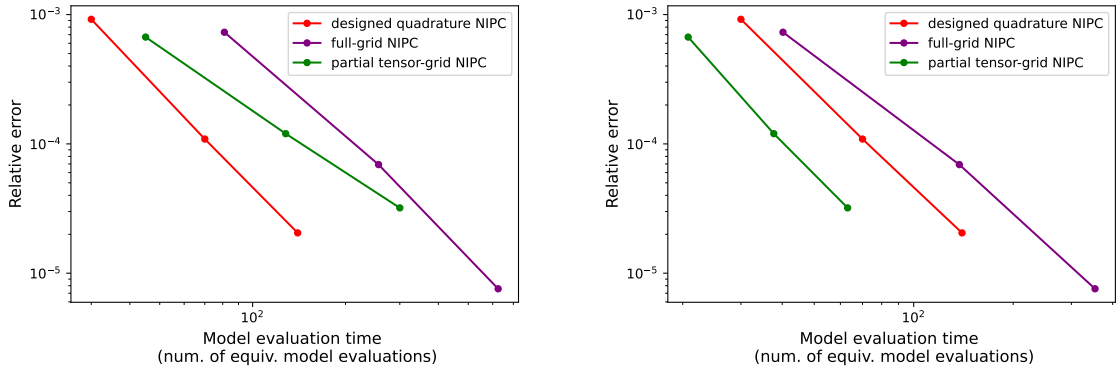
Table 4.3. Sparsity ratio of the uncertain inputs in the UAV model

Uncertain inputs	Sparsity ratio
v	92%
h	96%
W_p	4%
σ	2%

inputs are identified: W_p and σ, as both of them have a sparsity ratio less than 5%. Combining this information with our understanding of the computational model, we constructed the partially tensor-structured quadrature rule as:

$$u = u^{n_1}_{\{h,v\}} \times u^{n_2}_{\{W_p\}} \times u^{n_3}_{\{\sigma\}}. \tag{4.38}$$

Our proposed method is compared with the designed quadrature method and the full-grid Gauss quadrature method, all of which are used with the NIPC method to solve the UQ problem. The UQ convergence plots, with and without using AMTC, are shown in Fig. 4.4. From the results, we observe that the partially tensor-structured quadrature rule is outperformed by the designed quadrature method when AMTC is not used to accelerate the model evaluations. This finding aligns with our theoretical analysis, as the non-tensorial quadrature rule requires fewer quadrature points than the tensor-structured quadrature rules to achieve the same level of accuracy. However, when we utilize AMTC to accelerate the model evaluations, the advantages of the structured quadrature rules become evident. Both the model evaluations associated with the full-grid quadrature rule and the partially tensor-structured quadrature rule are significantly accelerated by using the AMTC method. In this scenario, the partially tensor-structured quadrature rule achieves the best performance, with more than 50% reduction in evaluation costs than the other two methods. This is because, our method constructed a tailored tensorial structure that makes the most effective use of the inherent sparsity within the computational graph of this multidisciplinary system, resulting in the lowest model evaluation costs under the same level of accuracy.



(a) Convergence plots without AMTC **(b)** Convergence plost with AMTC

Figure 4.4. Convergence plots with and without AMTC for the UAV problem

4.3.1 6-dimensional UQ problem with an air-taxi trajectory design multidisciplinary model

The second test problem is a 6D UQ problem adapted from an air-taxi trajectory optimization scenario. This computational model calculates the average ground-level sound pressure level during the aircraft's flight along a specific trajectory. The UQ problem aims to determine the expected output under uncertainties in the control inputs and parameters of the acoustic model. Three uncertain inputs are associated with the initial conditions of the trajectory, and the other three uncertain inputs are associated with the parameters in the acoustic model. The distributions of the uncertain inputs can be found in Tab. 4.4. Notably, the control inputs used in this problem are generated from solving a deterministic trajectory optimization problem, which aims to minimize the total energy expended throughout the trajectory. Details of the computational model along with the trajectory optimization formulation can be found in [71].

Table 4.4. Input parameters and ranges for the air-taxi problem

Uncertain inputs	Distributions
Initial velocity v_0 (m/s)	$\mathscr{U}(30,35)$
Initial flight path angel γ_0 (deg)	$\mathscr{U}(0,2)$
Initial altitude h_0 (m)	$\mathscr{U}(0,20)$
β_1	$\mathscr{N}(0.0209,0.001)$
β_2	$\mathscr{N}(18.2429,0.5)$
β_3	$\mathscr{N}(6.729,0.2)$

Table 4.5. Sparsity ratio of the uncertain inputs in the air-taxi model

Uncertain inputs	Sparsity ratio
v_0	99%
h_0	99%
γ_0	99%
β_1	2%
β_2	2%
β_3	2%

In this problem, the measured sparsity ratios are shown in Tab. 4.5. Based on this information, three uncertain inputs are identified as sparse uncertain inputs which are the three uncertain inputs associated with the acoustic model. This finding aligns with the multidisciplinary structure of the computational model, depicted in Fig. 5.4. From this figure, we observe that the computational model comprises four disciplines: flight dynamics, aerodynamics, propulsion, and acoustics, with two-way coupling among the first three disciplines. In this case, the coupled disciplines functions as a nonlinear operation and is solved iteratively. Due to its dominant role in computational costs, the three uncertain inputs associated with the nonlinear solver are the dense uncertain inputs in this computational model. Conversely, the parameter uncertain inputs are the sparse uncertain inputs as they do not affect the upstream nonlinear solver in this computational model. Based on this reasoning, we choose the partially tensor-structured option as

$$u = u^{n_1}_{\{v_0,h_0,\gamma_0\}} \times u^{n_2}_{\{\beta_1,\beta_2,\beta_3\}}, \tag{4.39}$$

in which we form a tensor structure between the space of dense uncertain inputs and the space of sparse uncertain inputs.

The UQ convergence plots with and without using AMTC are shown in Fig. 4.6. Similar to the first test problem, the partially tensor-structured quadrature rule is outperformed by the designed quadrature method when AMTC is not used. However, it becomes the most efficient method when using AMTC to accelerate the model evaluations. The reduction in computational costs is more than 40% in most cases. This can be explained by the fact that the tailored tensorial

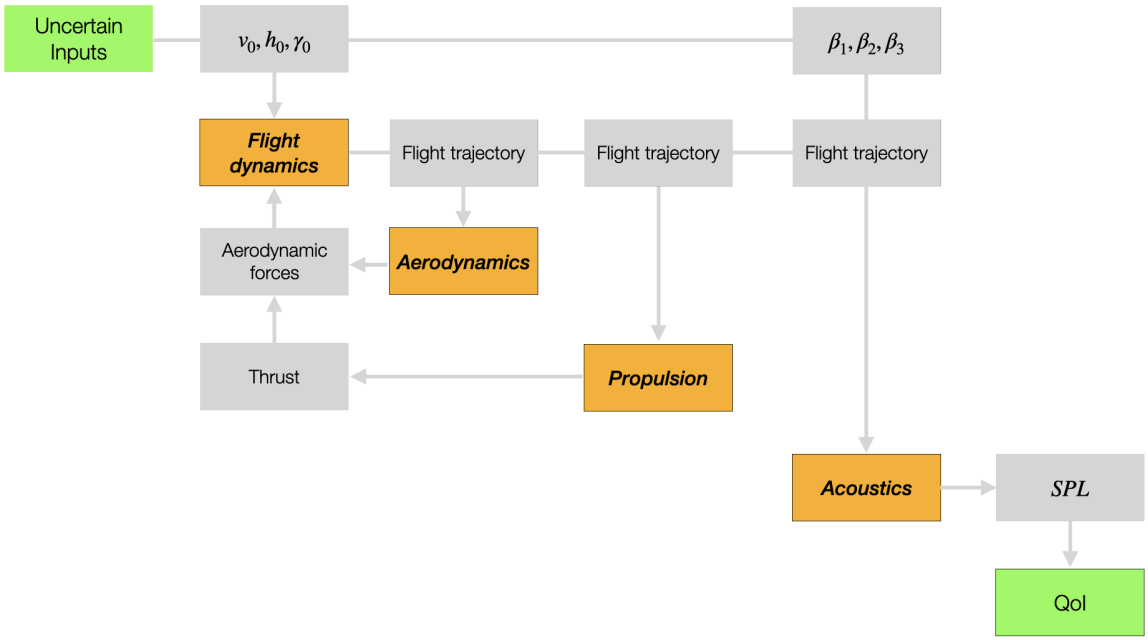
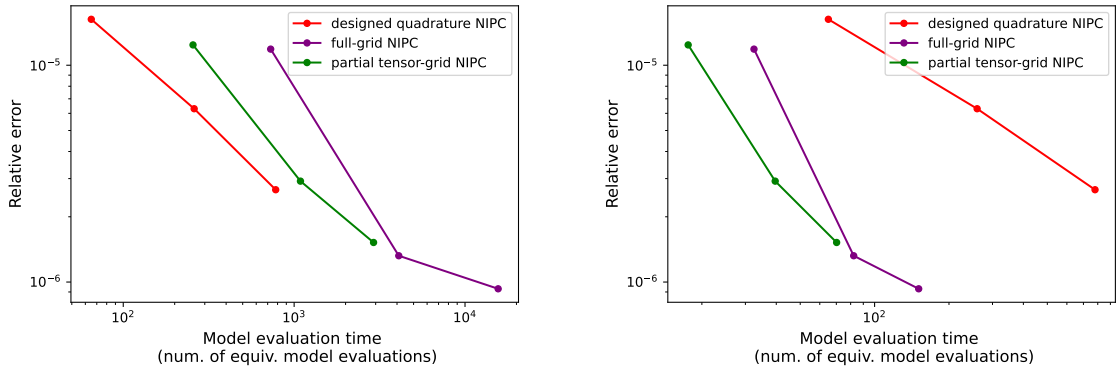


Figure 4.5. Multidisciplinary structure of the air-taxi model

structure option takes full advantage of the inherent sparsity of the computational graph, resulting in the most efficient performance with the AMTC method. However, we also observe that when the desired relative error is exceptionally low (1e-6 and lower), the performance of the partially tensor-grid quadrature rule deteriorates on this test problem. This phenomenon can be attributed to the fact that the designed quadrature rule method can achieve similar performance to the Gauss quadrature rule while requiring a significantly smaller number of model evaluations. However, as the dimension of the optimization problem increases (as we have more optimization variables and more constraints), it becomes more challenging to solve the optimization problem to a tight tolerance using this method. Consequently, the designed quadrature method may struggle to achieve the same level of accuracy as the Gauss quadrature rule when an extremely high level of accuracy is required.



(a) Convergence plot without AMTC **(b)** Convergence plot with AMTC

Figure 4.6. Convergence plots with and without AMTC for the 6D air-taxi problem

Acknowledgments

The material presented in this chapter is, in part, based upon work supported by DARPA under grant No. D23AP00028-00.

Chapter 4, in part, has been submitted for publication of the material [98] as it may appear in Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J. T., "Graph-accelerated non-intrusive polynomial chaos expansion using partially tensor-structured quadrature rules," Aerospace Science and Technology. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Extension of graph-accelerated NIPC to high-dimensional UQ through the active subspace method

The graph-accelerated NIPC methods introduced in Chapter 3 and Chapter 4 combine a novel computational graph transformation method, AMTC, with the integration-based NIPC method to tackle UQ problems with multidisciplinary systems. The AMTC method accelerates tensor-grid evaluations by modifying the computational graph of the model, thereby eliminating redundant evaluations at the operation level caused by the tensor structure of the inputs. By connecting it to the integration-based NIPC and generating the tensor-structured quadrature rules, we can exploit sparsities in the computational models. While this approach has demonstrated superiority over existing UQ methods for specific low-dimensional UQ problems involving multidisciplinary systems, it suffers from the curse of dimensionality in many high-dimensional (ten or more dimensions) UQ problems.

When addressing high-dimensional UQ problems, employing dimension-reduction techniques can sometimes significantly reduce computational costs. Notable methods include sensitivity analysis [67], principal component analysis (PCA) [2], partial least squares (PLS) [17], and active subspace (AS) [18]. Sensitivity analysis is a commonly used method that determines the impact of input parameters on output uncertainty. Local sensitivity analysis measures the perturbation of each input at a nominal value and its effect on output, making it inexpensive

to implement. However, this method may lack accuracy in many cases. In contrast, global sensitivity analysis [91] computes Sobol indices, which measure output variations over the full range of inputs, providing more accurate results but at a higher computational cost. The inputs corresponding to the highest sensitivities are then selected to reduce the dimension of the problems. PCA conducts eigenvalue decomposition on the covariance matrix of uncertain inputs, revealing the highest-variance directions in the input space. Yet, PCE is not well-suited for independent uncertain inputs. PLS utilizes regression to identify covariant directions between inputs and outputs. However, it can suffer from overfitting and does not leverage gradient information. On the other hand, AS uses gradient information to identify an active subspace where most of the first-order changes in the output exist. AS strikes a balance between accuracy and computational cost, making it a powerful tool for high-dimensional UQ problems, while also providing an intuitive way to visualize and interpret the results.

In this chapter, we aim to extend the capabilities of the graph-accelerated NIPC methods to solve high-dimensional UQ problems by leveraging the active subspace method. While the graph-accelerated NIPC method has proven effective in solving low-dimensional UQ problems with sparsity in computational graphs, there is currently no existing framework that can directly connect this method to the AS method for solving high-dimensional UQ problems. In previous works, Glaws and Constantines connected the AS method to the Gaussian quadrature rules to solve integration problems [30], but this method is limited to one-dimensional active subspace. Additionally, in [35], He et al., linked the AS method to regression-based NIPC methods, but this approach cannot be used to connect the active subspace method to the graph-accelerated NIPC method, which relies on an integration-based approach to solve the polynomial chaos expansion (PCE) coefficients. To address this gap, this paper first proposes a general framework called AS-NIPC, which generates PCE basis functions for the active subspace's uncertain inputs and an efficient quadrature rule in the original uncertain input space to connect the active subspace method to the integration-based NIPC method. This framework can be applied to any high-dimensional UQ problem without using the AMTC method. Furthermore, an extension of this

framework, AS-AMTC, is presented to use with the AMTC method. This extension involves identifying the sparse uncertain inputs (uncertain inputs that only affect a small amount of the computational cost), applying the AS method to non-sparse uncertain inputs, and generating a desired tensor-structured quadrature rule to take advantage of the computational graph sparsity when the AMTC method is applicable.

The proposed methods have been tested on two UQ problems which include a 7-dimensional analytical piston model and an 81-dimensional air-taxi trajectory simulation model. The results show that the proposed AS-NIPC is more effective than the existing methods for both of the problems while AS-AMTC can further improve its efficiency on the second problem which involves a multidisciplinary system.

5.1 Active subspace

The active subspace (AS) method, proposed by Constantine et. al. in [18], is a dimension reduction method that has recently gained a lot of popularity. The basic idea behind AS is to exploit the fact that some directions in the input parameter space have a greater impact on the output uncertainty than others. By identifying these important directions, we can project the high-dimensional input space onto a lower-dimensional subspace that captures the most important first-order input-output relationships. To achieve that, the AS method computes the mean squared gradients of the objective function with respect to the uncertain inputs as

$$C = E[(\nabla_u f)(\nabla_u f)^T] = \int_\Gamma (\nabla_u f)(\nabla_u f)^T \rho(u) du, \tag{5.1}$$

where $\nabla_u f = [\frac{\partial f}{\partial u_1}, \ldots, \frac{\partial f}{\partial u_d}]^T$. The C matrix is symmetric and positive semi-definite and thus has non-negative eigenvalues and orthogonal eigenvectors. Following an eigenvalue decomposition, the C matrix can be expressed as

$$C = W \Lambda W^T, \quad \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_d). \tag{5.2}$$

The eigenvectors W define a rotation of the original uncertain input space and are sorted by the magnitude of their corresponding eigenvalues in decreasing order, while the magnitude of the eigenvalues represents the function's average variation in each direction. The AS method identifies the active subspace $\tilde{u}$ by partitioning the eigenvalues and eigenvectors as

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \quad W = \begin{bmatrix} W_1 & W_2 \end{bmatrix}, \tag{5.3}$$

where $\Lambda_1 = \text{diag}(\lambda_1, \ldots, \lambda_m)$ and $W_1 \in R^{m \times d}$ with $m < d$. This results in two sets of rotated coordinates defined as

$$\tilde{u} = W_1^T u, \quad \bar{u} = W_2^T u, \tag{5.4}$$

such that the function is significantly more variant in the space of $\tilde{u}$ than $\bar{u}$. Once the active subspace is identified, the UQ problem can be solved in this lower-dimensional space, which greatly reduces the computational cost of the analysis.

5.2 Methodology

In this paper, we first propose a novel framework to connect the integration-based NIPC method to the active subspace approach to solve high-dimensional UQ problems. This involves generating orthogonal PCE basis functions of the active subspace variables using the whitening matrix and generating an efficient quadrature rule in the original input space using the designed quadrature method to estimate the PCE coefficients. We then propose an extension of this framework to combine it with the AMTC method to generate the quadrature rule that has a desired tensor structure to take advantage of the sparsity of the computational graph.

5.2.1 AS-based NIPC

Consider a high-dimensional UQ problem involving a function,

$$f = \mathscr{F}(u) \quad u \in R^d, \tag{5.5}$$

with the independent uncertain inputs $U = (U_1, \ldots, U_d) \in R^d$. We first apply the active subspace (AS) method to find a desired rotation of the input coordinate such that the function is only significantly variant with respect to a small number of inputs. This involves first computing the mean squared gradients of the objective function with respect to the uncertain inputs as

$$C = E[(\nabla_u f)(\nabla_u f)^T], \tag{5.6}$$

followed by an eigenvalue decomposition as

$$C = W \Lambda W^T, \quad \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_d). \tag{5.7}$$

The eigenvalues and eigenvectors are then partitioned as

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \quad W = \begin{bmatrix} W_1 & W_2 \end{bmatrix} \tag{5.8}$$

with $\Lambda_1 = \text{diag}(\lambda_1, \ldots, \lambda_m)$. This separates the rotated coordinates into two sets as

$$\tilde{u} = W_1^T u, \quad \bar{u} = W_2^T u, \tag{5.9}$$

and the function varies significantly more with respect to $\tilde{u}$ than $\bar{u}$. We call $\tilde{u}$ the *active variables* and $\bar{u}$ the *inactive variables*. Now the function can be written as

$$f(\tilde{u},\bar{u}) = \mathscr{F}(W_1 u + W_2 u). \tag{5.10}$$

By using the gPC theory and truncating all of the terms of $\bar{u}$, we approximate the model output as

$$f(\tilde{U}) \approx \sum_{i=0}^{q} \alpha_i \Phi_i(\tilde{U}). \tag{5.11}$$

Since $\tilde{u}$ can be significantly lower dimensional than the original uncertain input space, u, solving the PCE coefficients in equation (5.11) can be significantly less expensive than directly applying NIPC. However, applying the integration-based NIPC method here poses two main challenges:

1. The probability density of the distribution of the uncertain inputs in the active space is unknown which makes it difficult to generate the PCE terms in the active subspace.

2. An efficient quadrature rule in the original input space needs to be derived to compute the PCE coefficients that correspond to the PCE terms of the active subspace uncertain inputs.

Generating PCE basis functions

We solve the first challenge by computing the whitening matrices that represent the coefficients of the PCE basis functions. We follow the method in [57] and first represent the PCE basis functions of $\tilde{u}$ as

$$\Phi(\tilde{u}) = \left[\Phi_0(\tilde{u}),\ldots,\Phi_q(\tilde{u})\right]^T = MP_k(\tilde{u}), \tag{5.12}$$

where $P_k(\tilde{u})$ is the monomial basis vector in $\tilde{u}$ up to degree k, and $M \in R^{(k+1)\times(k+1)}$ is a lower triangular matrix storing the coefficients for the monomial basis polynomials in the univariate PCE basis functions. For demonstration, in a two-dimensional case, the first six Legendre

polynomials (PCE basis functions for uniform uncertain inputs) are

$$\begin{aligned}
\Phi_0(u) &= 1; \\
\Phi_1(u) &= u_1; \\
\Phi_2(u) &= u_2; \\
\Phi_3(u) &= u_1 u_2; \\
\Phi_4(u) &= \frac{1}{2}(3u_1^2 - 1); \\
\Phi_5(u) &= \frac{1}{2}(3u_2^2 - 1).
\end{aligned} \tag{5.13}$$

The PCE basis functions in (5.13) can be represented in the form of the monomial basis vector as

$$\underbrace{\begin{bmatrix} \Phi_0(u) \\ \Phi_1(u) \\ \Phi_2(u) \\ \Phi_3(u) \\ \Phi_4(u) \\ \Phi_5(u) \end{bmatrix}}_{\Phi(u)} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 & \frac{3}{2} & 0 \\ -\frac{1}{2} & 0 & 0 & 0 & 0 & \frac{3}{2} \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} 1 \\ u_1 \\ u_2 \\ u_1 u_2 \\ u_1^2 \\ u_2^2 \end{bmatrix}}_{P_2(u)} \tag{5.14}$$

We want to solve for the M matrix so that the orthogonality property in (2.6) is satisfied in terms of $\tilde{u}$ as

$$\int_{\Gamma_{\tilde{u}}} \Phi(\tilde{u})\Phi(\tilde{u})^T \rho(\tilde{u}) d\tilde{u} = I, \tag{5.15}$$

which can be written as

$$\int_{\Gamma_{\tilde{u}}} P_k(\tilde{u})P_k(\tilde{u})^T \rho(\tilde{u}) d\tilde{u} = M^{-1}M^{-T}. \tag{5.16}$$

We define the monomial matrix as

$$G := \int_{\Gamma_{\tilde{u}}} P_k(\tilde{u})P_k(\tilde{u})^T \rho(\tilde{u}) d\tilde{u}, \tag{5.17}$$

then the M matrix satisfies

$$M^{-1}M^{-T} = G, \tag{5.18}$$

and can be solved through Cholesky factorization of G as

$$M = Q^{-1}, \quad G = QQ^T. \tag{5.19}$$

However, solving G in (5.17) would require us to know the probability distribution of the active variables, $\rho(\tilde{u})$. To avoid solving $\rho(\tilde{u})$, we utilize the linear relationship, $\tilde{u} = W_1^T u$ and apply the change of variables to rewrite (5.17) as

$$\begin{aligned} G &= \int_{\Gamma_{\tilde{u}}} P_k(\tilde{u}) P_k(\tilde{u})^T \rho(\tilde{u}) d\tilde{u} \\ &= E[P_k(\tilde{u}) P_k(\tilde{u})^T] \\ &= E[P_k(W_1^T u) P_k(W_1^T u)^T] \\ &= \int_{\Gamma_u} P_k(W_1^T u) P_k(W_1^T u)^T \rho(u) du. \end{aligned} \tag{5.20}$$

The integral in (5.20) does not require any model evaluation and can then be easily computed by using a Monte Carlo or numerical quadrature approach.

Generating an efficient quadrature rule

Using the integration-based NIPC approach, we approximate the PCE coefficients in (5.11) by solving an integration problem as

$$\begin{aligned} \alpha_i &= \frac{\left\langle f(\tilde{U}), \Phi_i(\tilde{u}) \right\rangle}{\langle \Phi_i(\tilde{u})^2 \rangle} \\ &= \int_{\Gamma_{\tilde{u}}} f(\tilde{u}) \Phi_i(\tilde{u}) \rho(\tilde{u}) d\tilde{u}. \end{aligned} \tag{5.21}$$

This results in an integration problem in $\tilde{u}$ space, which is much lower dimensional than the original uncertain input space. However, evaluating this integral is challenging in two parts.

Firstly, evaluating $f(\tilde{u})$ directly is not possible as the computational model only accepts inputs in terms of u. Moreover, for a given point in the $\tilde{u}$ space, there may exist an infinite number of points in u that satisfy $\tilde{u} = W_1^T u$. Secondly, the probability density distribution of the active variables, denoted as $\rho(\tilde{u})$, is unknown. Consequently, direct generation of quadrature points is not feasible. The objective here is to generate an efficient quadrature rule with the nodes in the original input space. We denote the quadrature points and the weights as $u := (u^{(1)}, \ldots, u^{(n)})$ and $w := (w^{(1)}, \ldots, w^{(n)})$ respectively, and the quadrature rule approximates the integral in (5.21) as

$$\int_{\Gamma_{\tilde{u}}} f(\tilde{u})\Phi_i(\tilde{u})\rho(\tilde{u})d\tilde{u} \approx \sum_{i=1}^{d} w^{(i)} f(u^{(i)})\Phi_i\left(W_1^T(u^{(i)})\right). \tag{5.22}$$

We solve the quadrature rule following the designed quadrature method in [54]. The designed quadrature method generates the quadrature rule by solving an optimization problem to ensure they can exactly integrate polynomial functions up to a specific order. The core idea here is that for an integration problem

$$I(f) = \int_{\Gamma} f(u)\rho(u)du, \tag{5.23}$$

if the nodes and weights of the quadrature rule satisfy the multi-variate moment matching equations as

$$\sum_{i=1}^{n} \Phi_{i'}\left(u^{(i)}\right) w^{(i)} = \begin{cases} 1 & \text{if } |i'| = 0 \\ 0 & \text{for } \alpha \in 0 < |i'| < k, \end{cases} \tag{5.24}$$

where $\Phi_{i'}$ are the PCE basis functions of u, then the quadrature rule can exactly integrate any polynomials of u up to $(2k-1)$th order. In our case, we want to design the quadrature rule with u and w such that after the linear transformation, $\tilde{u} = W_1^T u$, the quadrature rule can exactly integrate polynomials of $\tilde{u}$ to a specific order. Thus we write the moment-matching equation here as

$$\sum_{i=1}^{n} \Phi_{i'}\left(W_1^T u^{(i)}\right) w^{(i)} = \begin{cases} 1 & \text{if } |i'| = 0 \\ 0 & \text{for } 0 < |i'| < k, \end{cases} \tag{5.25}$$

where $\Phi_{i'}$ are the PCE basis functions of $\tilde{u}$ that we generated in the previous step. We represent the moment matching equations in (5.25) as residual equations:

$$\mathscr{R}(u,w) = 0, \tag{5.26}$$

where $\mathscr{R} : R^{d\times n} \times R^n \to R^n$. The quadrature rule is generated by solving an optimization problem formulated as

$$\begin{aligned} &\min_{u,w} \|r(u,w)\|_2 \\ &\text{subject to } u^{(j)} \in \Gamma, \quad j = 1,\ldots,n, \\ &\qquad\qquad w^{(j)} > 0, \quad j = 1,\ldots,n, \end{aligned} \tag{5.27}$$

which minimizes the norm of the residual equations and constrains the nodes to be within the support range and weights to be positive. The number of nodes to use, n, depends on the dimension of the active subspace, m, and the highest degree of polynomials of PCE basis functions, k, and can be chosen following [99], as

$$n = \begin{cases} \frac{k^m}{m} & \text{for } m \leq 2 \\ 0.9\frac{(2m)^{k-1}}{(k-1)!} & \text{for } m > 2. \end{cases} \tag{5.28}$$

The optimization problem in (5.27) can be solved using any common nonlinear optimizer.

Special case: Gaussian uncertain inputs

One special case in which we can easily generate the PCE basis functions is when all of the uncertain inputs are Gaussian random variables. The rotated coordinates $\tilde{u}$ satisfy a linear relationship with the original coordinates u as $\tilde{u} = W_1^T u$. When the uncertain inputs are mutually independent Gaussian random inputs, since the eigenvectors in W_1 are orthogonal, the random variables associated with the active variables are also mutually independent random variables

and their distributions can be represented as

$$\tilde{U} \sim \mathcal{N}(W_1^T \mu, W_1^T \Sigma W_1), \tag{5.29}$$

where μ and Σ are the mean vector and covariance matrix associated with the original uncertain inputs U, respectively. This means that the PCE basis functions of $\tilde{u}$ are simply the multivariate Hermite polynomials that can be easily generated. For the quadrature rule, we still recommend using the same designed quadrature approach, as this approach results generally require the fewest quadrature points to achieve the same level of accuracy. Alternatively, if the dimension of the uncertain inputs is small enough ($m \leq 3$), one may choose to use the full-grid quadrature rule in $\tilde{u}$ and solve the quadrature points in the original input space such that $W_1^T u = \tilde{u}$.

AS-NIPC algorithm

We refer to this AS-based NIPC method as the AS-NIPC method and present a detailed step-by-step algorithm:

1. **Discover the active subspace:** Apply the AS method to discover the active variables $\tilde{u}$, following (5.6)(5.7)(5.8)(5.9).

2. **Generate the PCE basis functions:** Solve the whitening matrix, M, following (5.20) (5.19) (5.12) for a specified polynomial order k.

3. **Generate the quadrature rule:** Compute the nodes u and weights w in the quadrature rule by solving the optimization problem in (5.27), choose the number of quadrature points, n, using (5.28).

4. **Compute the PCE coefficients:** Evaluating the model/function on the quadrature points and computing each PCE coefficient following (5.21)(5.22).

5. **Compute the QoIs:** Computing the desired quantity of interests of the model output following the NIPC method.

5.2.2 AS based NIPC with AMTC

The AS-NIPC method we presented in the previous section provides a framework to combine the integration-based NIPC method with the active subspace method to solve high-dimensional UQ problems. However, this method cannot be accelerated by the AMTC method as the quadrature points do not possess a tensor structure. The objective here is to propose an extension of the AS-NIPC method to generate the quadrature points that possess a desired tensor structure such that the model evaluations can be significantly accelerated using the AMTC method. We follow the partially tensor-structured quadrature rule in [99] to generate a desired tensor-structured quadrature rule to use in the graph-accelerated NIPC method.

For high-dimensional UQ problems, there often exist some uncertain input operations that affect a small number of operations in the computational graph. In this case, the evaluation cost of the operations that are dependent on these inputs may only take a small percentage of the total model evaluation cost. We refer to these uncertain inputs as *sparse uncertain inputs* and the other uncertain inputs as *non-sparse uncertain inputs*. The sparse uncertain inputs can be identified by computing the sparsity ratio (SR) of each uncertain input. SR(u_i) is defined as the ratio of the evaluation cost of the entire model to that of only the operations that are influenced by random input u_i and can be estimated based on the computational graph analysis detailed in [99]. We choose the sparse uncertain inputs as its sparsity ratio is $< 5\%$, and we partition the uncertain inputs as

$$u = (u_s, u_{ns}), \tag{5.30}$$

where $u_s = (u_{s_1}, \ldots, u_{s_{d_1}}) \in R^{d_1}$ is the set of sparse uncertain inputs, and $u_{ns} \in R^{d_2}$ is the set of non-sparse uncertain inputs with $d = d_1 + d_2$. Since the sparse uncertain inputs only affect a small amount of model evaluation cost, we want to choose the quadrature rule that maintains a tensor structure between the quadrature points in the non-sparse uncertain input space and quadrature points in the space of each sparse uncertain input, such that the quadrature points can

be written as

$$u = u_{\{ns\}}^{n_1} \times u_{\{s_1\}}^{n_2} \times \ldots, u_{\{s_{d_1}\}}^{n_2}, \tag{5.31}$$

where $u_{\{ns\}}^{n_1}$ represents quadrature points in u_{ns} space with n_1 points and $u_{\{s_{d_1}\}}^{n_2}$ represent the quadrature points in u_{s_i} space with n_2 points. This results in a total of $n_1 n_2^{d_2}$ quadrature points in the quadrature rule. However, employing the AMTC method to transform the computational graph leads to a computational cost on the modified graph that scales roughly linearly with n_1. This is because most operations in the computational graph depend solely on u_{ns} and are evaluated only n_1 times. However, for high-dimensional UQ problems, the dimensions for the non-sparse uncertain inputs can still be large enough such that the required model evaluation cost is still unaffordable even with the AMTC method. We address this problem by applying the AS-NIPC method within the space of u_{ns} to generate an efficient quadrature rule with respect to the active variables in the space non-sparse uncertain inputs. This involves computing the C matrix as

$$C = E[(\nabla_{u_{ns}} f)(\nabla_{u_{ns}} f)^T] = \int_\Gamma (\nabla_{u_{ns}} f)(\nabla_{u_{ns}} f)^T \rho(u) du, \tag{5.32}$$

then the AS method can be used to find the active variables that satisfy

$$\tilde{u}_{ns} = W_1^T u_{ns}, \tag{5.33}$$

such that $\tilde{u}_{ns} \in R^m$ with $m \leq d_1$. Following the AS-NIPC method, we can generate the PCE basis functions of the active variables, written as $\Phi(\tilde{U}_{\{ns\}})$ as well as the quadrature rule with the nodes in U_{ns} space, we denote the nodes as $u_{ns}^{\tilde{n}}$. Using the NIPC approach, we can approximate the stochastic output as the PCE basis functions of the active variables and the non-sparse uncertain inputs, as

$$f(\tilde{U}_{ns}, U_s) \approx \sum_{i=0}^{q} \alpha_i \Phi_i(\tilde{U}_{ns}, U_s), \tag{5.34}$$

where the PCE basis functions can be constructed as

$$\Phi(\tilde{U}_{ns}, U_s) = \Phi(\tilde{U}_{ns}) \times \phi(u_{s_1}) \times \ldots \times \phi(u_{s_{d_2}}), \tag{5.35}$$

where $\phi(u_{s_i})$ represent the univariate orthogonal polynomials of u_{s_i}. The quadrature rule follows the same tensor structure and the quadrature points can be written as

$$u = u^{\tilde{n}}_{\{ns\}} \times u^{k}_{\{s_1\}} \times \ldots, u^{k}_{\{s_{d_1}\}}, \tag{5.36}$$

where the $u^{k}_{\{s_i\}}$ is chosen as the k Gauss quadrature points in u_{s_i} dimension. This results in an efficient quadrature rule with a desired tensor structure to use with the AMTC method when it comes to model evaluations.

AS-AMTC algorithm

We refer to this AS-based NIPC method as the AS-NIPC method and present a detailed step-by-step algorithm:

1. **Identify the sparse uncertain inputs:** Identify the sparse uncertain inputs that only affect a small amount of the computational cost.

2. **Apply AS-NIPC:** Apply AS-NIPC in the non-sparse uncertain inputs space. Generate the PCE basis functions of the active variables as well as the quadrature rule in the non-sparse uncertain inputs space.

3. **Form the tensor-structured quadrature rule:** Form the PCE basis functions and the quadrature rule in a tensor structured following (5.35) and (5.36), respectively.

4. **Model evaluations with AMTC:** Perform the model evaluations on the tensor-structured inputs using AMTC.

Table 5.1. Uncertain inputs and distributions for the piston problem

Uncertain input	Distribution	Description
M	$N(45,3)$	Piston weight (kg)
S	$N(0.01,0.001)$	Piston surface area (m^2)
V_0	$N(0.010,0.001)$	Initial gas volume (m^3)
k	$N(3000,200)$	Spring coefficient (N/m)
P_0	$N(90000,5000)$	Atmospheric pressure (N/m^2)
T_a	$N(290,20)$	Ambient temperature (K)
T_0	$N(340,20)$	Filling gas temperature (K)

5. **Compute the QoIs:** Compute the PCE coefficients and the desired quantity of interests of the model output.

5.3 Numerical Results

5.3.1 7-dimensional UQ problem with an analytical piston simulation model

The first test problem involves an analytical, nonlinear model that calculates the cycle time of a piston within a cylinder, adapted from [9]. The cycle time C of the piston, measured in seconds, is defined as:

$$C = 2\pi\sqrt{\frac{M}{k+S^2\frac{P_0V_0T_a}{T_0V^2}}}, \tag{5.37}$$

where V is given by:

$$V = \frac{S}{2k}\left(\sqrt{A^2+4k\frac{P_0V_0}{T_0}T_a}-A\right) \tag{5.38}$$

and A is calculated as

$$A = P_0S + 19.62M - \frac{kV_0}{S}. \tag{5.39}$$

The UQ problem aims to determine the expected value of the piston cycle time considering seven uncertain inputs, detailed in Table 5.1.

This problem is solved using three UQ methods: Monte Carlo, Active-Subspace-based kriging (AS-kriging), and AS-NIPC. The UQ result using Monte Carlo with 100,000 sample points is regarded as the ground truth result. AS-kriging is implemented by randomly sampling

Table 5.2. Normalized eigenvalues of the C matrix in the piston problem

AS-kriging & AS-NIPC
1.00
3.82e-16
1.53e-21
1.13e-25
1.68e-25
9.71e-27
2.17e-30

the active variables according to their distributions and constructing the corresponding response surface function. To assess their performance, the relative errors of AS-kriging and Monte Carlo methods are averaged over 20 runs. For AS-NIPC and AS-kriging, the approximation of the C matrix is computed using 100 sample points, and the resulting eigenvalues are presented in Table 5.2. In this problem, we notice a rapid decline in eigenvalues after the first one, leading us to select a one-dimensional active subspace ($m = 1$). The convergence plots for the three UQ methods are depicted in Figure 5.1. It is evident from these plots that both AS-based approaches outperform the Monte Carlo method significantly when constrained to 10s or fewer model evaluations. This improvement is attributed to the discovery of a 1D active subspace, effectively reducing the UQ problem's dimension from 7 to 1. When comparing AS-kriging and AS-NIPC methods, both tend to reach a relative error limit around $5e-3$. However, AS-NIPC achieves this level of accuracy with considerably fewer evaluations, owing to NIPC's effectiveness in handling low-dimensional UQ problems. Specifically, in this problem, AS-NIPC with only 5 model evaluations results in a lower error than AS-kriging with 20 model evaluations.

5.3.2 81-dimensional UQ problem with an air-taxi trajectory simulation model

The second test problem is an 81-dimensional UQ problem involving a lift-plus-cruise electric air taxi trajectory simulation model. The representation of the aircraft is shown in Fig. 5.2. This computational model calculates the average ground-level sound pressure level

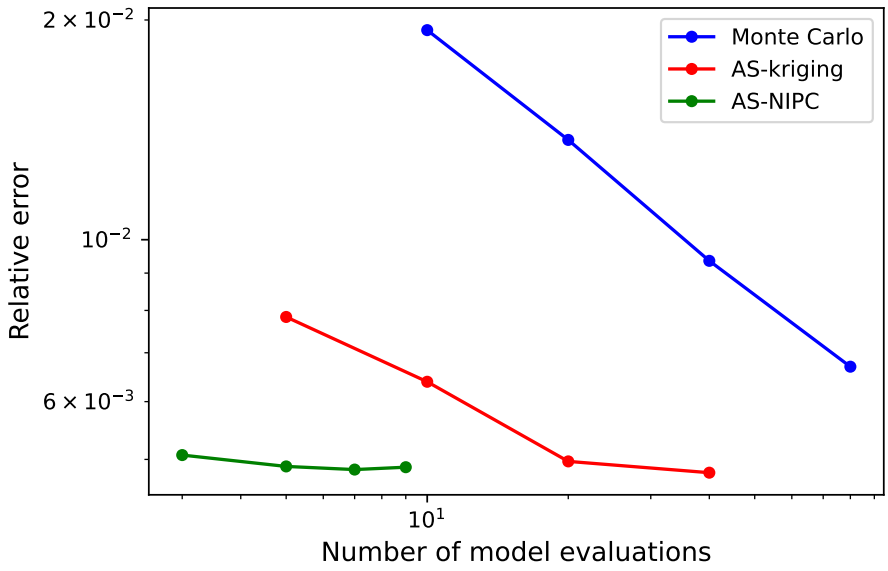


Figure 5.1. UQ results on the 7D piston problem

during the aircraft's flight along a specific trajectory [71]. The UQ problem aims to determine the expected output under uncertainties in the control inputs and acoustic parameters. The model is composed of two sub-models: the trajectory model and the acoustic model. The trajectory model integrates three disciplines—flight dynamics, aerodynamics, and propulsion—with two-way coupling among them. It computes the aircraft's flight path based on the control inputs history, including the histories of lift and cruise rotor thrust (x_l, x_c). On the other hand, the acoustics model computes the ground-level sound pressure level (SPL) based on the flight trajectory, using a correlation equation with three parameters: $(\beta_1, \beta_2, \beta_3)$. Further details about these models can be found in [71]. The multidisciplinary structure of the model is shown in 5.4.

In this UQ problem, we have control inputs with 40 time steps, denoted as $x_l = [x_{l_0}, \ldots, x_{l_{39}}]$ and $x_c = [x_{c_0}, \ldots, x_{c_{39}}]$. These inputs' average history is denoted as $\bar{x}_l = [\bar{x}_{l_0}, \ldots, \bar{x}_{l_{39}}]$ and $\bar{x}_c = [\bar{x}_{c_0}, \ldots, \bar{x}_{c_{39}}]$. We assume a linear relationship between consecutive steps of control inputs, with white noise added at each step, following:

$$\begin{aligned} X_{l_{i+1}} &= cX_{l_i} + N_{l_{i+1}}, \quad N_{l_{i+1}} \sim \mathcal{N}(0, 0.003\bar{x}_{l_i}); \\ X_{c_{i+1}} &= cX_{c_i} + N_{c_{i+1}}, \quad N_{c_{i+1}} \sim \mathcal{N}(0, 0.003\bar{x}_{c_i}). \end{aligned} \tag{5.40}$$

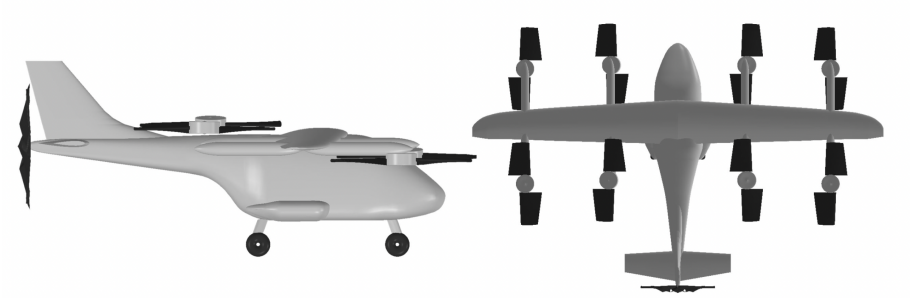
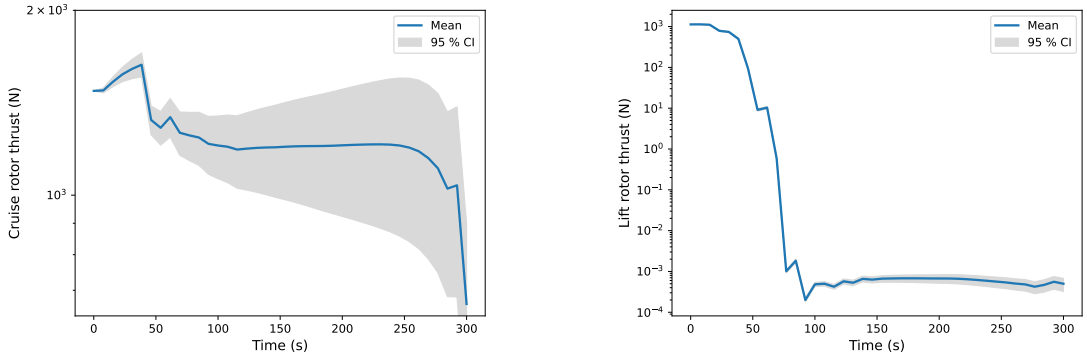


Figure 5.2. Representation of the lift-plus-cruise electric air taxi [71]



(a) Cruise rotor thrust

(b) Lift rotor thrust

Figure 5.3. Control inputs with confidence intervals

This formulation is commonly used to represent control input uncertainties in motion planning under uncertainty problems [44]. Fig. 5.3 illustrates the control inputs' histories alongside their averages and 95% confidence intervals. Additionally, we have Gaussian uncertain inputs associated with the three parameters in the acoustic model, $(\beta_1, \beta_2, \beta_3)$. The complete set of 81 uncertain inputs and their distributions are detailed in Tab. 6.2.

Table 5.3. Uncertain inputs in the UQ problem

Uncertain input	Distribution
$N_l \in R^{39}$	$\mathcal{N}(0, 0.003\bar{x}_l)$
$N_c \in R^{39}$	$\mathcal{N}(0, 0.003\bar{x}_c)$
$\beta_1 \in R$	$\mathcal{N}(0.0209, 0.002)$
$\beta_2 \in R$	$\mathcal{N}(18.2429, 2)$
$\beta_3 \in R$	$\mathcal{N}(6.729, 0.7)$

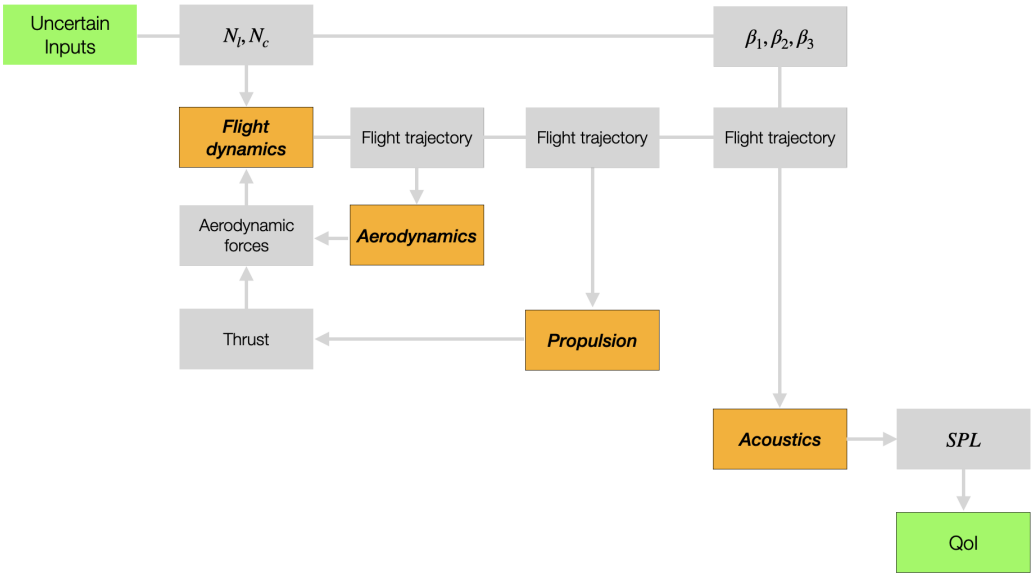


Figure 5.4. Multidisciplinary structure of the air-taxi model

This problem is solved by four UQ methods: Monte Carlo, AS-kriging, AS-NIPC, and AS-AMTC. Both the AS-kriging and AS-NIPC methods involve applying the AS method in the original uncertain input space. For these two methods, the C matrix in (5.6) is approximated using the Monte Carlo method with the same 100 sample points. We present the first seven eigenvalues of the C matrix in Tab. 5.4 and choose $m = 6$ as the dimension of the active subspace as we observe a rapid decay after the sixth eigenvalue. For the AS-AMTC method, we first compute the sparsity ratio of each uncertain input and the results are shown in Tab. 7.3. The results indicate that the parameter uncertain inputs, (β_1,β_2, β_3) are the sparse uncertain inputs in this computational model with the sparsity ratio of 2%. This observation can be attributed to the fact that these parameter uncertainties only affect the operations within the acoustic sub-model, which constitutes a very small portion of the overall computational cost. In AS-NIPC, the AS method is specifically employed within the non-sparse uncertain input space, which comprises only N_l and N_c. In Tab. 5.4, we present the first seven eigenvalues alongside those corresponding to the other two AS methods. Analyzing Tab. 5.4, we notice that the eigenvalues in the AS-AMTC method decay more rapidly compared to the other case, leading us to select an active

Table 5.4. Normalized eigenvalues (first seven) of the C matrix

AS-kriging & AS-NIPC	AS-AMTC
1.0000	1.0000
0.0661	0.0592
0.0649	0.0381
0.0459	0.0080
0.0346	0.0032
0.0194	0.0004
0.0032	0.0002

Table 5.5. Sparsity ratio of the uncertain inputs

Uncertain inputs	Sparsity ratio
N_l	99%
N_c	99%
β_1	2%
β_2	2%
β_3	2%

subspace dimension of $m = 5$.

The convergence plots of the four UQ methods are shown in Fig. 5.5. Notably, the Monte Carlo method exhibits superior performance when conducting 3000 or more model evaluations. This superiority stems from the inherent trade-off in dimension reduction methods, where reducing problem dimensions results in information loss about the inputs, limiting the dimension reduction-based UQ methods' ability to achieve extremely high accuracy. When compared to the AS-kriging method, both of our proposed methods demonstrate superior performance. For instance, with 100 function evaluations, AS-NIPC achieves a relative error at least 30% lower than AS-kriging, while AS-AMTC achieves a relative error at least 80% lower than AS-kriging. The exceptional performance of AS-AMTC can be attributed to the presence of three sparse uncertain inputs in the computational graph of the model. These inputs only affect about 2% of the model evaluation cost. In this case, maintaining a tensor structure of the quadrature rule with each of the sparse uncertain inputs is very effective to use with the AMTC method. Additionally, by applying the AS method exclusively to the uncertain input space excluding the sparse uncertain inputs, we observe a more rapid decay rate of the eigenvalues for the C matrix,

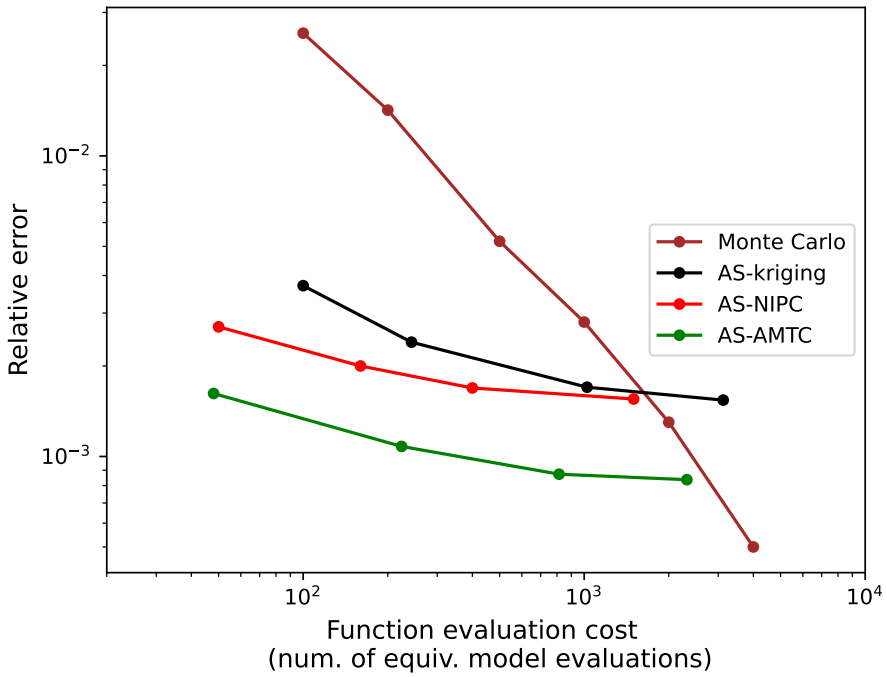


Figure 5.5. UQ results on the 81D air-taxi problem

as detailed in Tab. 5.4. This enables us to choose a smaller dimension for the active subspace while achieving a more accurate reduced-order model.

Acknowledgments

The material presented in this paper is, in part, based upon work supported by DARPA under grant No. D23AP00028-00 and by NASA under award No. 80NSSC21M0070.

Chapter 5, is taken, in part, from the material [101] which has been submitted for publication as Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J.T., "Extension of graph-accelerated non-intrusive polynomial chaos to high-dimensional uncertainty quantification through the active subspace method," Structural and Multidisciplinary Optimization, submitted. The dissertation author was the primary investigator and author of this paper.

Chapter 6

A gradient-enhanced univariate dimension reduction method for UQ

In the previous chapters, the graph-accelerated NIPC method has been introduced and has been applied to a broad range of UQ problems assuming the a certain type of sparsity is present in the computational graph of the original computational model. In some problems with multidisciplinary systems, the computational graph transformation method, AMTC, by taking advantage of the computational graph sparsity, enables a significant speed-up on tensor-grid evaluations. However, if the computational graph does not possess this type of sparsity, the graph-accelerated NIPC methods may be outperformed by other UQ methods.

This paper proposes a new UQ method, gradient-enhanced univariate dimension reduction (GUDR), which uses the AMTC method in a way that does not rely on the inherent sparsity within the computational graph. This method is based on a popular dimension reduction method, unvariate dimension reduction (UDR), and intends to enhance its accuracy in estimating second and higher-order statistical moments by incorporating univariate gradient terms into the UDR approximations.

This method has been applied to four UQ problems: one 2D and one 3D mathematical function, one 4D rotor aerodynamic analysis problem, and one 7D problem derived from a practical aircraft design scenario. The results indicate that on the mathematical functions, when estimating the standard deviation of the output, GUDR is more accurate than UDR and has

comparable performance as the method of moments method using third-order Taylor expansion. On the 4D rotor analysis and 7D aircraft design problems, GUDR enhances the accuracy of UDR by an order of magnitude in estimating the standard deviation and becomes the most cost-effective method to use among the UQ methods implemented.

6.1 Univariate dimension reduction

We consider an uncertainty quantification (UQ) problem involving a function $f(u)$, where $u \in R^d$ denotes the input vector, and $f \in R$ represents a scalar output. The uncertain inputs are expressed as a stochastic vector denoted by $U := [U_1, \ldots, U_d]^T$, with the assumption that these random variables are mutually independent. The stochastic input vector has probability density distribution $\rho(u)$ with support Γ. The UQ problem aims to compute the statistical moments of the output random variable, $f(U)$. Given that the mean of the uncertain inputs vector is $\mu := [\mu_1, \ldots, \mu_d]^T$, the univariate dimension reduction (UDR) [78] method approximates the original function $f(u)$ using a sum of univariate functions:

$$\hat{f}(u) = \sum_{i=1}^{d} f_i(u_i) - (d-1) f(\mu), \tag{6.1}$$

where each univariate function term is defined as $f_i(u_i) := f(\mu_1, \ldots, \mu_{i-1}, u_i, \mu_{i+1}, \ldots, \mu_d)$. Expanding $f(u)$ in a Taylor series at $u = \mu$ yields

$$\begin{aligned} f(u) = f(\mu) &+ \sum_{j=1}^{\infty} \sum_{i=1}^{d} \frac{1}{j!} \frac{\partial^j f}{\partial u_i^j}(\mu)(u_i - \mu_i)^j \\ &+ \sum_{j_2=1}^{\infty} \sum_{j_1=1}^{\infty} \frac{1}{j_1! j_2!} \sum_{i_1=1}^{d-1} \sum_{i_2>i_1}^{d} \frac{\partial^{j_1+j_2} f}{\partial u_{i_1}^{j_1} \partial u_{i_2}^{j_2}}(\mu)(u_{i_1} - \mu_{i_1})^{j_1}(u_{i_2} - \mu_{i_2})^{j_2} \\ &+ \sum_{j_3=1}^{\infty} \sum_{j_2=1}^{\infty} \sum_{j_1=1}^{\infty} \frac{1}{j_1! j_2! j_3!} \sum_{i_1=1}^{d-2} \sum_{i_2>i_1}^{d-1} \sum_{i_3>i_2}^{d} \frac{\partial^{j_1+j_2+j_3} f}{\partial u_{i_1}^{j_1} \partial u_{i_2}^{j_2} \partial u_{i_3}^{j_3}}(\mu) \\ &(u_{i_1} - \mu_{i_1})^{j_1}(u_{i_2} - \mu_{i_2})^{j_2}(u_{i_3} - \mu_{i_3})^{j_3} + \ldots \end{aligned} \tag{6.2}$$

The Taylor series expansion of the UDR approximation function, $\hat{f}(u)$, at $u = \mu$ can be expressed as

$$\hat{f}(u) = f(\mu) + \sum_{j=1}^{\infty} \sum_{i=1}^{d} \frac{1}{j!} \frac{\partial^j f}{\partial u_i^j}(\mu)(u_i - \mu_i)^j. \tag{6.3}$$

If we compare the Taylor series of the original function with the UDR approximation function, all of the Taylor series terms in (6.3) are contained in (6.2), and the residual errors are

$$\begin{aligned} f(u) - \hat{f}(u) = & \sum_{j_2=1}^{\infty} \sum_{j_1=1}^{\infty} \frac{1}{j_1! j_2!} \sum_{i_1=1}^{d-1} \sum_{i_2>i_1}^{d} \frac{\partial^{j_1+j_2} f}{\partial u_{i_1}^{j_1} \partial u_{i_2}^{j_2}}(\mu)(u_{i_1} - \mu_{i_1})^{j_1}(u_{i_2} - \mu_{i_2})^{j_2} \\ & + \sum_{j_3=1}^{\infty} \sum_{j_2=1}^{\infty} \sum_{j_1=1}^{\infty} \frac{1}{j_1! j_2! j_3!} \sum_{i_1=1}^{d-2} \sum_{i_2>i_1}^{d-1} \sum_{i_3>i_2}^{d} \frac{\partial^{j_1+j_2+j_3} f}{\partial u_{i_1}^{j_1} \partial u_{i_2}^{j_2} \partial u_{i_3}^{j_3}}(\mu) \\ & (u_{i_1} - \mu_{i_1})^{j_1}(u_{i_2} - \mu_{i_2})^{j_2}(u_{i_3} - \mu_{i_3})^{j_3} + \ldots \end{aligned} \tag{6.4}$$

This shows that the UDR approximation is only second-order accurate in approximating the original function [78]. However, when it comes to estimating the mean of the output, the residual errors become

$$\begin{aligned} E[f(U)] - E[\hat{f}(U)] &= \int_{\Gamma_1} \ldots \int_{\Gamma_d} \left(f(u) - \hat{f}(u)\right) \rho(u) du_1 \ldots du_d \\ &= \frac{1}{2!2!} \sum_{i_1=1}^{d-1} \sum_{i_2>i_1}^{d} \frac{\partial^4 f}{\partial u_{i_1}^2 \partial u_{i_2}^2}(\mu) E[(u_{i_1} - \mu_{i_1})^2 (u_{i_2} - \mu_{i_2})^2] + \ldots, \end{aligned} \tag{6.5}$$

which involves only fourth and higher-order integration terms. In contrast, the 3rd-order Taylor series method approximates the function as

$$\begin{aligned} \tilde{f}(u) = & f(\mu) + \sum_{j=1}^{3} \sum_{i=1}^{d} \frac{1}{j!} \frac{\partial^j f}{\partial u_i^j}(\mu)(u_i - \mu_i)^j \\ & + \sum_{j_1=1}^{2} \sum_{j_2=1}^{j_1+j_2 \le 3} \frac{1}{j_1! j_2!} \sum_{i_1=1}^{d-1} \sum_{i_2>i_1}^{d} \frac{\partial^{j_1+j_2} f}{\partial u_{i_1}^{j_1} \partial u_{i_2}^{j_2}}(\mu)(u_{i_1} - \mu_{i_1})^{j_1}(u_{i_2} - \mu_{i_2})^{j_2} \\ & + \sum_{i_1=1}^{d-2} \sum_{i_2>i_1}^{d-1} \sum_{i_3>i_2}^{d} \frac{\partial^3 f}{\partial u_{i_1} \partial u_{i_2} \partial u_{i_3}}(\mu)(u_{i_1} - \mu_1)(u_{i_2} - \mu_2)(u_{i_3} - \mu_3). \end{aligned} \tag{6.6}$$

When estimating the mean of the output with this method, the residual errors are

$$E[f(U)]-E[\tilde{f}(U)]=\frac{1}{2!2!}\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\frac{\partial^4 f}{\partial u_{i_1}^2\partial u_{i_2}^2}(\mu)E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2] + \sum_{i=1}^{d}\frac{1}{4!}\frac{\partial^4 f}{\partial u_i^4}(\mu)E[(u_i-\mu_i)^4]+\ldots \tag{6.7}$$

Upon comparing the UDR's residual errors in (6.5) with those of the 3rd-order Taylor series in (6.7), we observe that both methods are fourth-order accurate in estimating the mean of the output. However, the UDR comprises fewer fourth-order integration terms. If we assume the constants on the fourth-order terms are similar, it suggests that UDR generally yields a more accurate result than the 3rd-order Taylor expansion when estimating the mean of the output.

Nevertheless, for higher-order statistical moments, the UDR approximation may not perform as well as the 3rd-order Taylor expansion. For instance, when estimating the second-order statistical moments, the UDR's relative errors are:

$$\begin{aligned}E[f(U)^2]-E[\hat{f}(U)^2]=&\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\left(\frac{\partial^2 f}{\partial u_{i_1}\partial u_{i_2}}(\mu)\right)^2 E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]\\ &+\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\left(\frac{\partial f}{\partial u_{i_1}}(\mu)\frac{\partial^3 f}{\partial u_{i_1}\partial u_{i_2}^2}(\mu)+\frac{\partial f}{\partial u_{i_2}}(\mu)\frac{\partial^3 f}{\partial u_{i_1}^2\partial u_{i_2}}(\mu)\right)\\ &E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]\\ &+\frac{1}{2!}f(\mu)\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\frac{\partial^4 f}{\partial u_{i_1}^2\partial u_{i_2}^2}(\mu)E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]+\ldots\end{aligned} \tag{6.8}$$

In comparison, the residual errors for the 3rd order Taylor series method are

$$E[f(U)^2]-E[\tilde{f}(U)^2]=\frac{1}{2!}f(\mu)\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\frac{\partial^4 f}{\partial u_{i_1}^2\partial u_{i_2}^2}(\mu)E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]+\ldots \tag{6.9}$$

Comparing the residual errors of the 3rd-order Taylor series method in (6.9) with the residual errors of the UDR in (6.8), both of the methods are also fourth-order accurate in estimating the

second-order moment of the output, but the relative errors of the UDR comprise significantly more fourth-order integration terms compared with the 3rd-order Taylor series expansion. Thus it is expected that the UDR is not as accurate as the 3rd Taylor series expansion method when used to estimate the second order moment of the output.

The detailed derivation of the residual errors for the UDR approximation in estimating the first and second-order statistical moments for a 2D problem can be found in Appendix 6.4.1.

Computational cost

The major advantage of the UDR approximation is that when estimating the statistical moments of the output, the corresponding multidimensional integration problem can be decomposed into multiple one-dimensional integration problems, which significantly reduces the required computational cost. For example, when computing the mean of the output, the UDR yields

$$E[\hat{f}(U)] = \sum_{i=1}^{d} \int_{\Gamma_i} f_i(u_i)\rho(u_i)du_i - (d-1)f(\mu). \tag{6.10}$$

This only requires performing d one-dimensional integrations instead of performing a d-dimensional integration. If we use the quadrature rule to approximate each integral with k quadrature points, the total number of model evaluations required is $kd+1$, which scales only linearly with the dimension of the UQ problem. For higher-order statistical moments of the output, UDR can compute it following a recurring formula in [78] without requiring extra model evaluations.

With an efficient automatic differentiation method to compute the gradients, our method preserves UDR's linear scaling of computation time with problem dimension.

6.2 Methodology

6.2.1 Gradient-enhanced univariate dimension reduction

We now describe the new methodology that forms the primary contribution of this chapter. The key idea here is to postulate a new approximation expression that enhances the accuracy of the univariate dimension reduction (UDR) approximation expression by adding the univariate gradient terms. With the addition of these gradient terms, the new approximation expression can be more accurate in estimating the second and higher-order statistical moments while the required computational cost still scales linearly with the problem dimension. We coin the new method as the *gradient-enhanced univariate dimension reduction* (GUDR) method. GUDR approximates the original function, f, as

$$\begin{aligned}\bar{f}(u) = &\sum_{i=1}^{d} f_i(u_i) - (d-1)f(\mu) + \sum_{i=1}^{d}(u_i-\mu_i)\sum_{j\neq i}^{d}\frac{\partial f_j}{\partial u_i}(u_j) \\ &- (d-1)\sum_{i}^{d}(u_i-\mu_i)\frac{\partial f}{\partial u_i}(\mu) - \sum_{i=1}^{d}\sum_{j>i}^{d}\frac{\partial^2 f}{\partial u_i \partial u_j}(\mu)(u_i-\mu_i)(u_j-\mu_j),\end{aligned} \tag{6.11}$$

where $\frac{\partial f_j}{\partial u_i}(u_j) := \frac{\partial f}{\partial u_i}(\mu_1,\ldots,\mu_{j-1},u_j,\mu_{j+1},\ldots,\mu_d)$ represents a univariate gradient term. This form includes all of the terms in the UDR approximation function in (6.1), with additional terms including the univariate first-order gradient terms and the second-order derivatives evaluated at μ. The GUDR approximation can also be written in matrix form as

$$
\begin{aligned}
\bar{f}(x) = \sum_{i=1}^{d} f_i(u_i) - (d-1)f(\mu) + \sum_{i=1}^{d} \underbrace{\begin{bmatrix} u_1-\mu_1 \\ \vdots \\ u_{i-1}-\mu_{i-1} \\ 0 \\ u_{i+1}-\mu_{i+1} \\ \vdots \\ u_d-\mu_d \end{bmatrix}^T}_{(u-\mu)\odot(\mathbf{1}-\mathbf{e_i})} \underbrace{\begin{bmatrix} \frac{\partial f_i}{\partial u_1}(u_i) \\ \vdots \\ \frac{\partial f_i}{\partial u_d}(u_i) \end{bmatrix}}_{\frac{\partial f_i}{\partial u}(u_i)} \\
- (d-1) \underbrace{\begin{bmatrix} u_1-\mu_1 \\ \vdots \\ u_d-\mu_d \end{bmatrix}^T}_{u-\mu} \underbrace{\begin{bmatrix} \frac{\partial f}{\partial u_1}(\mu) \\ \vdots \\ \frac{\partial f}{\partial u_d}(\mu) \end{bmatrix}}_{\frac{\partial f}{\partial u}(\mu)} \\
- \underbrace{\begin{bmatrix} u_1-\mu_1 \\ \vdots \\ u_d-\mu_d \end{bmatrix}^T \begin{bmatrix} 0 & \frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu) & \dots & \frac{\partial^2 f}{\partial u_1 \partial u_d}(\mu) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \frac{\partial^2 f}{\partial u_{d-1} \partial u_d}(\mu) \\ 0 & \dots & \dots & 0 \end{bmatrix} \begin{bmatrix} u_1-\mu_1 \\ \vdots \\ u_d-\mu_d \end{bmatrix}}_{\frac{1}{2}(u-\mu)^T\left(\frac{\partial^2 f}{\partial u^2}(\mu) - \mathrm{diag}\left(\mathrm{diag}\left(\frac{\partial^2 f}{\partial u^2}(\mu)\right)\right)\right)(u-\mu)} \\
= \sum_{i=1}^{d} f_i(u_i) + \sum_{i=1}^{d} \left((u-\mu)\odot(\mathbf{1}-\mathbf{e_i})\right)^T \frac{\partial f_i}{\partial u}(u_i) - (d-1)\left(f(\mu) - (u-\mu)^T \frac{\partial f}{\partial u}(\mu)\right) \\
- \frac{1}{2}(u-\mu)^T \left(\frac{\partial^2 f}{\partial u^2}(\mu) - \mathrm{diag}\left(\mathrm{diag}\left(\frac{\partial^2 f}{\partial u^2}(\mu)\right)\right)\right)(u-\mu),
\end{aligned}
\tag{6.12}
$$

where $\frac{\partial f}{\partial u}(\mu)$ and $\frac{\partial^2 f}{\partial u^2}(\mu)$ represent the gradient vector and Hessian matrix evaluated at $u=\mu$, respectively. If we compare the Taylor series expansions at $u=\mu$ of the GUDR approximation function and the original function, just as with the UDR approximation, all of the Taylor series terms of the GUDR approximation function are included in the original function, and the residual

errors can be expressed as

$$f(u)-\bar{f}(u)=\sum_{i=1}^{d}\sum_{j>i}^{d}\sum_{k>j}^{d}\frac{\partial^3 f}{\partial u_i \partial u_j \partial u_k}(\mu)(u_i-\mu_i)(u_j-\mu_j)(u_k-\mu_k)+\ldots. \tag{6.13}$$

This shows the GUDR approximation is third-order accurate when used to approximate the original function. In comparison with the residual errors of the UDR approximation in (6.3), the GUDR approximation is a more accurate approximation of the original function as the UDR approximation is only second-order accurate.

When it comes to estimating the mean of the output, the GUDR approximation generates the same results as the UDR approximation with the residual errors expressed as

$$E[f(U)]-E[\bar{f}(U)]=\frac{1}{2!2!}\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\frac{\partial^4 f}{\partial u_{i_1}^2 \partial u_{i_2}^2}(\mu)E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]+\ldots. \tag{6.14}$$

This means that the UDR and GUDR approximations are expected to provide the same results in estimating the mean of the output, and both are expected to be more accurate than the 3rd-order Taylor series expansion. However, when it comes to estimating the higher-order statistical moments, the GUDR can provide more accurate results than UDR as the GUDR approximation is more accurate than the UDR approximation in estimating the original function. For example, when estimating the second-order statistical moment of the output, the residual errors of the GUDR approximation can be expressed as

$$E[f(U)^2]-E[\bar{f}(U)^2]=\frac{1}{2!}f(\mu)\sum_{i_1=1}^{d-1}\sum_{i_2>i_1}^{d}\frac{\partial^4 f}{\partial u_{i_1}^2 \partial u_{i_2}^2}(\mu)E[(u_{i_1}-\mu_{i_1})^2(u_{i_2}-\mu_{i_2})^2]+\ldots \tag{6.15}$$

When compared with the residual errors from the 3rd-order Taylor series expansion method, as detailed in (6.9), and those from the UDR method, seen in (6.8), all three methods exhibit residual errors that include only terms of the fourth order and higher. However, the residual errors of the GUDR and 3rd Taylor series expansion approximations contain the same fourth-order

integration terms, while the UDR contains significantly more fourth-order integration terms. This means that, when estimating the second-order statistical moments, the GUDR approximation is expected to have a comparable level of accuracy with the 3rd-order Taylor series expansion, and its result can be significantly more accurate than the UDR approximation.

The detailed derivation of the residual errors for the GUDR approximation in estimating the first and second-order statistical moments for a 2D problem can be found in Appendix 6.4.2.

6.2.2 Tensor-grid evaluations to estimate statistical moments

For a d-dimensional UQ problem, the GUDR approximation in (6.12) involves multiple univariate function terms and univariate gradient terms. When it comes to estimating the mean of the output, all of the gradient terms become zero, and the GUDR yields the same result as UDR,

$$E[\bar{f}(U)] = \sum_{i=1}^{d} \int_{\Gamma_i} f_i(u_i)\rho(u_i)du_i - (d-1)f(\mu), \tag{6.16}$$

which can be evaluated following the UDR method. However, when it comes to estimating higher-order statistical moments of the output, there is no easy way to decompose the multidimensional integrals involved. One may want to derive the recursive formula for GUDR following the UDR method. However, the recursive formula will be significantly more complicated and is unpractical to implement in real problems.

In this paper, we propose a new approach to decompose the GUDR approximation function from the computational-graph-transformation perspective, so that we efficiently generate the model evaluations on full-grid quadrature points. This approach is inspired by the recently developed computational graph transformation method, *Accelerated Model evaluations on Tensor-grid using Computational graph transformation* (AMTC).

If we treat each univariate function and gradient term in the GUDR approximation as individual operations, the computational graph of the GUDR approximation function is shown in Fig. 6.1. Despite the complicated form of the GUDR approximation, the main computational

cost only comes from evaluating the univariate function and gradient terms. When applying the GUDR approximation function to tensor-grid inputs, which are described as

$$u = u_1^k \times \ldots \times u_d^k, \tag{6.17}$$

where u_i^k denotes the set of k quadrature points within the u_i dimension, we note that despite the presence of k^d input points, only k distinct points exist within each input dimension. Drawing on AMTC's foundational concept, since every univariate function and gradient computation operation in the GUDR's computational graph is dependent on just one uncertain input, it is only necessary to evaluate these operations at the distinct quadrature points within their input space.

Consequently, by integrating the AMTC strategy, the modified computational graph for the GUDR approximation function's tensor-grid evaluations is shown in Fig. 6.2. In this way, we can generate the model evaluations of GUDR approximation function on tensor-grid inputs, $\bar{f}(u) \in \mathscr{R}^{k^d}$, with only k evaluations on each univariate function and gradient evaluation term.

We note that the AMTC method has only been implemented in the CSDL compiler and can only be applied to computational models that are built in the CSDL language. Fortunately, in this specific scenario, we can achieve the minimum number of evaluations in univariate function and gradient evaluation terms by manually adding the *Einsum*[1] operations. First, we evaluate the univariate model function and gradient function on the corresponding quadrature points in its input space. Then we manually add *Einsum* operations in the code to transform these quadrature points evaluations to the correct size with the correct order of the data. Lastly, this data is passed to the GUDR approximation function to generate the full-grid evaluations of the output. We show example codes to achieve this in the Appendix 6.4.3.

The tensor-grid input-output data can be easily used with the quadrature rule to compute any order of statistical moments of the output. Additionally, this data can also be used with other UQ methods like non-intrusive polynomial chaos, stochastic collocation, and kriging to construct

[1]Einsum refers to the Einstein summation function in Numpy.

a surrogate model to approximate the risk measures of the output, such as the probability of failure or conditional value at risk.

6.2.3 Computational cost

Utilizing the AMTC strategy enables the computation of any statistical moment or risk measure of the output with a fixed number of evaluations—specifically, k evaluations for each univariate function and its gradient term, accompanied by a single function evaluation, gradient evaluation, and Hessian evaluation at the mean of the input variables. With reverse-mode automatic differentiation, for a function with d inputs, a single gradient vector evaluation, and a full Hessian matrix evaluation carry the computational weight less than 3 and $3d$ model evaluations [16], respectively. As a result, a conservative estimation of the model evaluation cost for solving the d-dimensional UQ using GUDR with k quadrature points in each dimension of the uncertain input is no more than $4kd + 3d + 4$ model evaluations, which scales only linearly with the dimension of the problem.

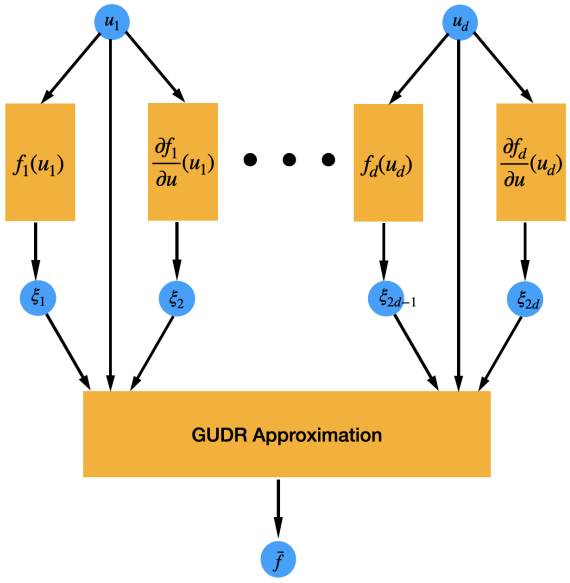


Figure 6.1. Computational graph of the GUDR approximation function

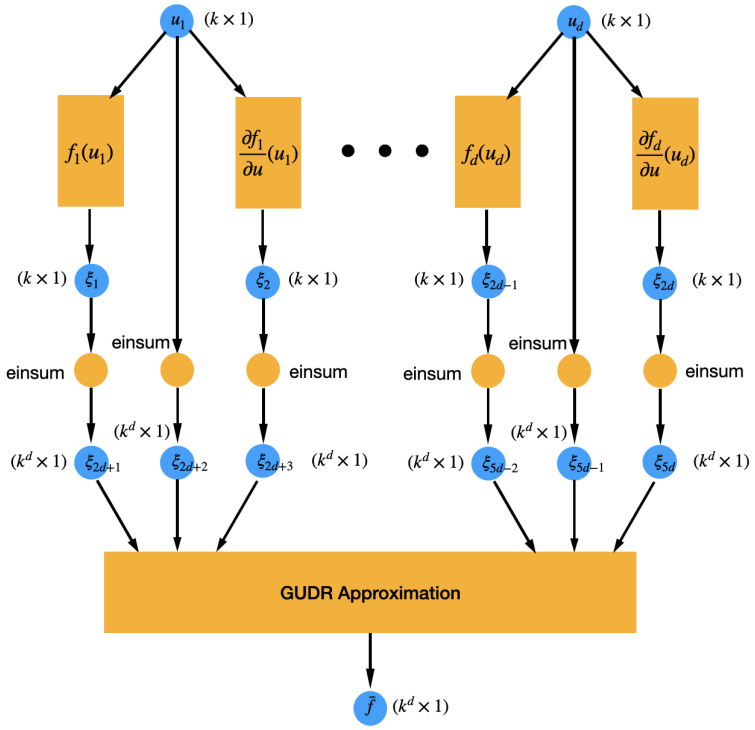


Figure 6.2. Computational graph of tensor-grid evaluations after graph transformation

6.3 Numerical Results

In Sec 6.3.1, we compare the accuracy of four UQ methods for estimating the standard deviation of the output: gradient-enhanced univariate dimension reduction (GUDR), univariate dimension reduction (UDR), method of moments using a 2nd order Taylor series expansion, and method of moments using a 3rd order Taylor series expansion. These methods are applied to two UQ problems involving different mathematical functions. In Sec 6.3.2 and Sec 6.3.3, we compare the convergence plots of GUDR, UDR, and the Monte Carlo methods in estimating both the mean and the standard deviation of the output on a 4D UQ problem involving a rotor aerodynamic analysis model and a 7D UQ problem involving an aircraft design multidisciplinary model, respectively.

6.3.1 2-dimensional and 3-dimensional UQ problems with mathematical functions

We examine two UQ problems that involve mathematical functions defined by simple, closed-form expressions. The first problem involves the function,

$$y_1 = f(x_1, x_2) = \frac{1}{1 + x_1^4 + 2x_2^2 + x_2^4} \tag{6.18}$$

with uncertain inputs,

$$X_i \sim \mathcal{N}(2, \sigma), \quad i = 1, 2. \tag{6.19}$$

The second problem involves the function:

$$y_2 = f(x_1, x_2, x_3) = \exp(1 + 0.5x_1^2 + 0.5x_2^2 + 0.5x_3^2) \tag{6.20}$$

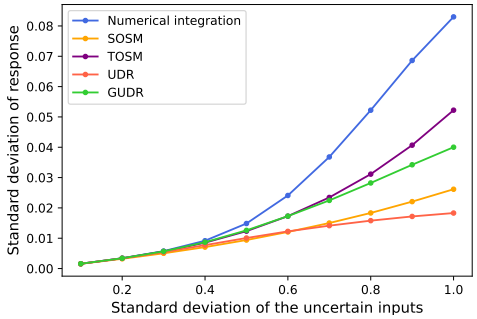
with uncertain inputs:

$$X_i \sim \mathcal{N}(3, \sigma), \quad i = 1, 2, 3. \tag{6.21}$$

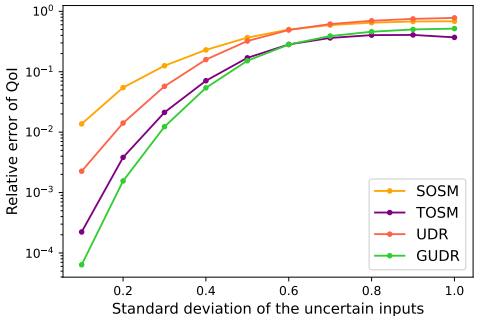
The quantity of interest (QoI) for both UQ problems is the standard deviation of the output Y_1 and Y_2, respectively. These problems are adapted from [78].

For these test problems, we have implemented five methods: UDR, GUDR, method of moments using the 2nd-order Taylor series expansion (second-order second-moment (SOSM)), method of moments using the 3rd-order Taylor series expansion (third-order second-moment (TOSM)), and direct numerical integration. The UDR and GUDR methods are implemented using 19 quadrature points in each dimension of the uncertain inputs to yield the most accurate UQ results based on these two analytical approximation expressions. Conversely, the direct numerical integration method utilizes the Monte Carlo method with 100,000 sample points, with its UQ results considered as the ground truth.

The UQ results derived from these five methods, along with the relative errors of the



(a) QoI results for increasing values of input standard deviation



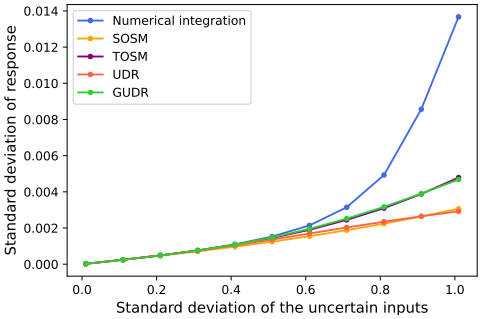
(b) Relative errors for increasing values of input standard deviation

Figure 6.3. UQ results for $y_1 = (1 + x_1^4 + 2x_2^2 + 5_2^4)^{-1}$

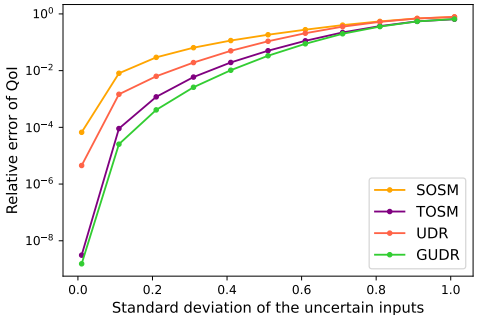
four analytical expression methods, are plotted against the increasing values of input standard deviations in Fig. 6.3 and Fig. 6.4 for Y_1 and Y_2, respectively. Our observations indicate that for both problems, all of the UDR, GUDR, SOSM, and TOSM methods yield accurate results for the standard deviation of the response when the standard deviation of the uncertain inputs is small. As we increase the standard deviation, the errors for all four methods correspondingly escalate. While the performances of UDR and SOSM are comparable, GUDR and TOSM also have comparable performances and consistently outperform UDR and SOSM. Typically, the relative errors of GUDR are less than those of UDR and SOSM by more than an order of magnitude. This superior performance of GUDR can be attributed to its more accurate approximation of the original function in comparison to UDR and the 2nd-order Taylor expansion. Additionally, the comparable performances of GUDR and TOSM match our theoretical results showing that the GUDR approximation function is expected to have comparable performance as the 3rd-order Taylor expansion when estimating the standard deviation of the output.

6.3.2 4-dimensional UQ problem with rotor aerodynamic analysis model

The third test problem is a 4D UQ problem involving the aerodynamic analysis of a rotor on an aircraft. The computational model applies the blade element momentum (BEM) theory to compute the rotor performance metrics including the thrust and torque generated. The



(a) QoI results for increasing values of input standard deviation



(b) Relative errors for increasing values of input standard deviation

Figure 6.4. UQ results for $y_2 = e^{(1+0.5x_1^2+0.5x_2^2+0.5x_3^2)}$

streamtube analyzed by the BEM model is shown in Fig. 6.5. Further details of the model can be found in [81]. The objective of the UQ problem is to determine the mean and standard deviation of the generated torque, given four uniform uncertain inputs shown in Tab. 6.1.

In this and the following test problems, we implemented four UQ methods: GUDR, UDR, non-intrusive polynomial chaos (NIPC), and the Monte Carlo method. The computational models were implemented in the Computational System Design Language (CSDL) [26]. The first-order gradient evaluations required for GUDR were calculated using an automatically implemented adjoint method detailed in [92], and the Hessian matrix was estimated via a finite difference method. The NIPC results were generated using the software package Chaospy [22], using a regression-based approach. We use the UQ results generated from the Monte Carlo method with $10,000$ sample points as the reference results and depict the convergence plots of the four methods in Fig. 6.6.

Table 6.1. Input parameters and ranges for the rotor analysis problem

Uncertain inputs	Distributions
Rotor speed (RPM)	$U(1600, 1800)$
Axial free-stream velocity (m/s)	$U(50, 60)$
Propeller radius (m)	$U(0.79, 0.81)$
Flight altitude (m)	$U(9000, 10000)$

From the results, it is evident that at a low evaluation cost (10s of equivalent model

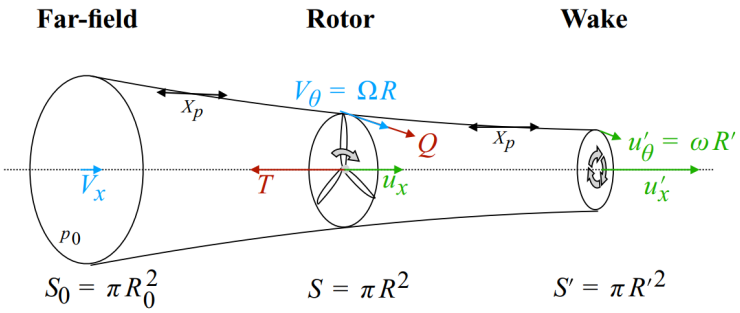
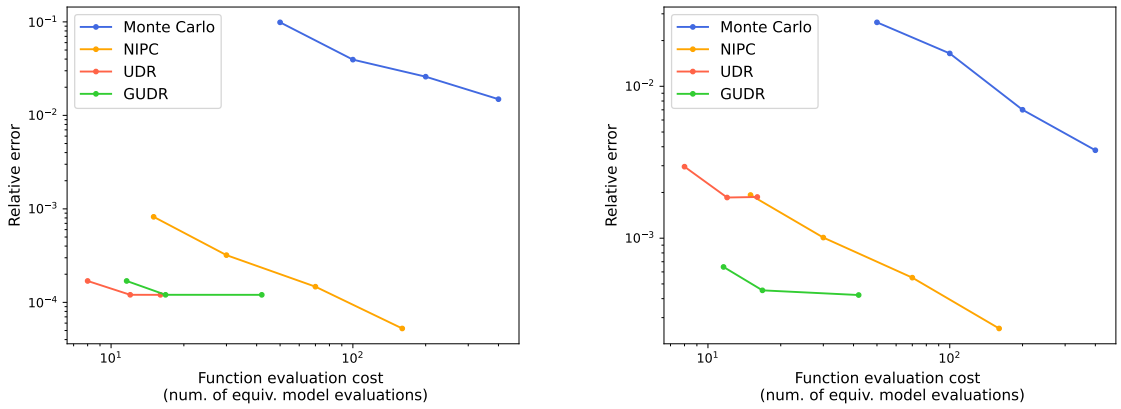


Figure 6.5. Streamtube for the blade element momentum model [81]



(a) Mean of the output

(b) Standard deviation of the output

Figure 6.6. Convergence plots for the 4D rotor analysis problem

evaluations), UDR, GUDR, and NIPC are capable of delivering more accurate results compared to the Monte Carlo method in terms of both mean and standard deviation for this particular problem. In terms of estimating the mean of the output, UDR and GUDR yield identical UQ results when employing the same number of quadrature points in each dimension, but GUDR is computationally more demanding than UDR. This observation is aligned with our theoretical analysis, which affirms that both GUDR and UDR are fourth-order accurate and produce identical results when estimating the mean of the output. Additionally, both UDR and GUDR are more accurate than NIPC at low evaluation costs (10s of equivalent model evaluations).

As for the standard deviation of the output, UDR's accuracy is comparable to NIPC, while GUDR enhances the accuracy of UDR by almost an order of magnitude at the expense of a slightly increased computational cost. This phenomenon can be attributed to the fact that

the GUDR approximation function results in fewer relative errors than the UDR approximation function when estimating the second and higher-order statistical moments. In this problem, with 20 equivalent model evaluations, GUDR can achieve a relative error of less than 0.1% for both mean and standard deviation of the output, making the GUDR method the most affordable UQ method to achieve this level of performance.

6.3.3 7-dimensional UQ problem with UAV design multidisciplinary model

The fourth test problem is a 7D UQ problem derived from a practical aircraft design scenario. This problem involves a high-altitude, laser beam-powered unmanned aerial vehicle (UAV) cruising at a high altitude while receiving charge from a laser beam emitted by a ground station. The concept of the UAV is illustrated in Fig. 6.7, while the laser beam engagement scenario is depicted in Fig. 6.8. The computational model incorporates multiple disciplines, such as aerodynamics, structures, thermals, power beaming, performance, and weight models. This model calculates the required aircraft mass to meet certain criteria, including endurance, maximum stress, and static margin. The meshes for the aerodynamics and structure solvers are shown in Fig. 7.1. For further details on the computational model, we refer the reader to [72].

The objective of the UQ problem is to determine the mean and standard deviation of the required aircraft mass, taking into account seven uncertain inputs with uniform distributions. The specific uncertain inputs and their distributions are listed in Tab. 6.2. The convergence plots of the four UQ methods are shown in Fig. 6.10.

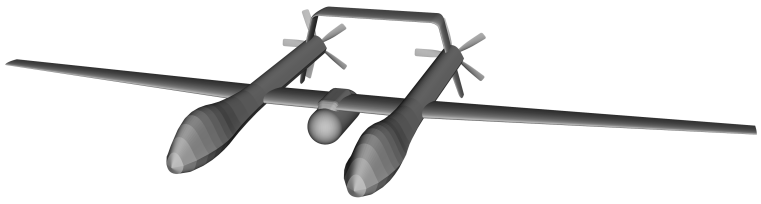


Figure 6.7. The high-altitude, laser-powered airplane concept

Similar to the last test problem, the Monte Carlo method is outperformed by UDR,

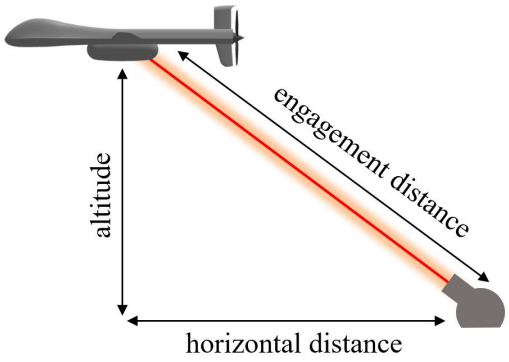
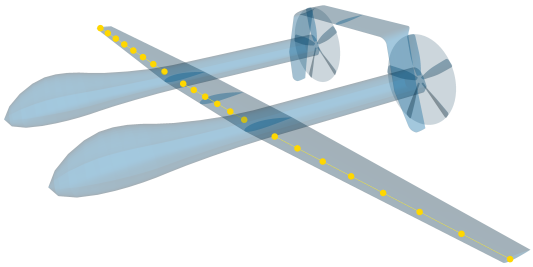
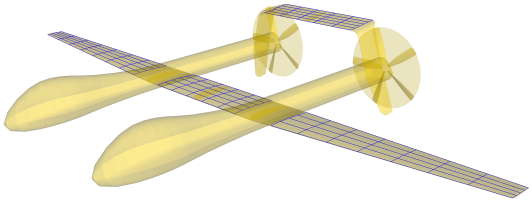


Figure 6.8. The laser beam engagement scenario



(a) 1D beam meshes for the structure solver



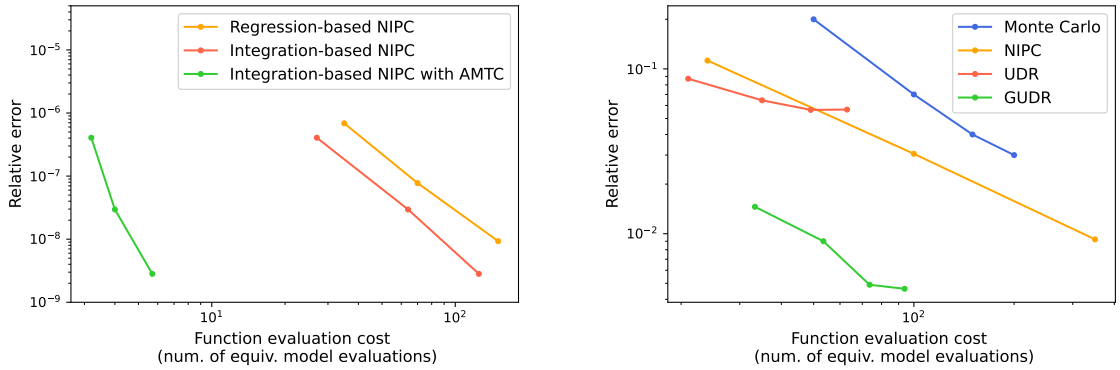
(b) 2D aerodynamics meshes for the aerodynamics solver

Figure 6.9. Meshes for structure and aerodynamics solvers

Table 6.2. Input parameters and ranges for the aircraft design problem

Uncertain inputs	Distributions
Cruise mach number	$U(0.1, 0.2)$
Altitude(km)	$U(17, 19)$
Horizontal distance (km)	$U(250, 300)$
Pitch angle (deg)	$U(-2, 2)$
Payload mass (kg)	$U(50, 100)$
Aperture diameter (m)	$U(0.7, 0.8)$
Rotor speed (RPM)	$U(1200, 1500)$

GUDR, and NIPC at low evaluation costs. When estimating the mean of the output, GUDR is slightly more expensive than UDR and both of them are more accurate than NIPC. However, when estimating the standard deviation of the output, UDR has comparable performance to NIPC at low function evaluation costs. In contrast, GUDR enhances the result of UDR by almost an order of magnitude with a slight increase in computational cost. These observations are also



(a) Mean of the output

(b) Standard deviation of the output

Figure 6.10. Convergence plots for the 7D aircraft design problem

consistent with our theoretical results as GUDR can be significantly more accurate than UDR in estimating second and higher-order statistical moments of the output. As a result, in this problem, with the computational cost of 50 equivalent model evaluations, GUDR can achieve a relative error of less than 1% for both the mean and standard deviation of the output, making the GUDR method the most affordable UQ method for achieving this level of performance.

6.4 Chapter appendix

6.4.1 Univariate dimension reduction in 2D case

In this section, we derive the residual errors of the univariate dimension reduction (UDR) method in estimating the mean and standard deviation for a 2D UQ problem. Consider an UQ problem that involves a bivariate function $f(u_1, u_2)$, the Taylor series expansion of f at $(u_1 = \mu_1, u_2 = \mu_2)$ can be expressed by

$$
\begin{aligned}
f(u_1,u_2) = f(\mu_1,\mu_2) &+ \frac{\partial f}{\partial u_1}(\mu_1,\mu_2)(u_1-\mu_1) \\
&+ \frac{\partial f}{\partial u_2}(\mu_1,\mu_2)(u_2-\mu_2) + \frac{1}{2!}\frac{\partial^2 f}{\partial u_1^2}(\mu_1,\mu_2)(u_1-\mu_1)^2 \\
&+ \frac{1}{2!}\frac{\partial^2 f}{\partial u_2^2}(\mu_1,\mu_2)(u_2-\mu_2)^2 + \frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2) \\
&+ \frac{1}{3!}\frac{\partial^3 f}{\partial u_1^3}(\mu_1,\mu_2)(u_1-\mu_1)^3 + \frac{1}{3!}\frac{\partial^3 f}{\partial u_2^3}(\mu_1,\mu_2)(u_2-\mu_2)^3 \\
&+ \frac{1}{2!}\frac{\partial^3 f}{\partial u_1^2 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2) + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^2 \\
&+ \frac{1}{4!}\frac{\partial^4 f}{\partial u_1^4}(\mu_1,\mu_2)(u_1-\mu_1)^4 + \frac{1}{4!}\frac{\partial^4 f}{\partial u_2^4}(\mu_1,\mu_2)(u_2-\mu_2)^4 \\
&+ \frac{1}{3!}\frac{\partial^4 f}{\partial u_1^3 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^3(u_2-\mu_2) + \frac{1}{3!}\frac{\partial^4 f}{\partial u_1 \partial u_2^3}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^3 \\
&+ \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2)^2 + \ldots
\end{aligned}
\tag{6.22}
$$

In this case, the UDR method approximates the original function as

$$
\hat{f}(u_1,u_2) = f(u_1,\mu_2) + f(\mu_1,u_2) - f(\mu_1,\mu_2). \tag{6.23}
$$

The Taylor series expansion of the UDR approximation, $\hat{f}$, at $(u_1 = \mu_1, u_2 = \mu_2)$ can be expressed as

$$
\begin{aligned}
\hat{f}(u_1,u_2) = f(\mu_1,\mu_2) &+ \frac{\partial f}{\partial u_1}(\mu_1,\mu_2)(u_1-\mu_1) + \frac{\partial f}{\partial u_2}(\mu_1,\mu_2)(u_2-\mu_2) \\
&+ \frac{1}{2!}\frac{\partial^2 f}{\partial u_1^2}(\mu_1,\mu_2)(u_1-\mu_1)^2 + \frac{1}{2!}\frac{\partial^2 f}{\partial u_2^2}(\mu_1,\mu_2)(u_2-\mu_2)^2 \\
&+ \frac{1}{3!}\frac{\partial^3 f}{\partial u_1^3}(\mu_1,\mu_2)(u_1-\mu_1)^3 + \frac{1}{3!}\frac{\partial^3 f}{\partial u_2^3}(\mu_1,\mu_2)(u_2-\mu_2)^3 \\
&+ \frac{1}{4!}\frac{\partial^4 f}{\partial u_1^4}(\mu_1,\mu_2)(u_1-\mu_1)^4 + \frac{1}{4!}\frac{\partial^4 f}{\partial u_2^4}(\mu_1,\mu_2)(u_2-\mu_2)^4 + \ldots
\end{aligned}
\tag{6.24}
$$

If we compare the Taylor series expansion of the original function with the UDR approximation, all of the Taylor series terms in (6.24) are contained in (6.22), and the residual errors can be

expressed as

$$
\begin{aligned}
f(u_1,u_2)-\hat{f}(u_1,u_2) = & \frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2) \\
& + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1^2 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2) \\
& + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^2 \\
& + \frac{1}{3!}\frac{\partial^4 f}{\partial u_1^3 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^3(u_2-\mu_2) \\
& + \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2)^2 \\
& + \frac{1}{3!}\frac{\partial^4 f}{\partial u_1 \partial u_2^3}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^3 + \ldots
\end{aligned}
\tag{6.25}
$$

From (6.25), we observe that the UDR approximation is only second-order accurate compared to the original function. However, when it comes to estimating the mean of the output, the residual errors can be expressed as

$$
\begin{aligned}
E[f(U_1,U_2)]-E[\hat{f}(U_1,U_2)] &= \int_{\Gamma_1}\int_{\Gamma_2}\left(f(u_1,u_2)-\hat{f}(u_1,u_2)\right)\rho(u_1)\rho(u_2)du_1du_2 \\
&= \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2] + \ldots
\end{aligned}
\tag{6.26}
$$

In comparison, for a 3rd-order Taylor series method which approximates the function as

$$
\begin{aligned}
\tilde{f}(u_1,u_2) = & f(\mu_1,\mu_2) + \frac{\partial f}{\partial u_1}(\mu_1,\mu_2)(u_1-\mu_1) + \frac{\partial f}{\partial u_2}(\mu_1,\mu_2)(u_2-\mu_2) \\
& + \frac{1}{2!}\frac{\partial^2 f}{\partial u_1^2}(\mu_1,\mu_2)(u_1-\mu_1)^2 + \frac{1}{2!}\frac{\partial^2 f}{\partial u_2^2}(\mu_1,\mu_2)(u_2-\mu_2)^2 \\
& + \frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2) + \frac{1}{3!}\frac{\partial^3 f}{\partial u_1^3}(\mu_1,\mu_2)(u_1-\mu_1)^3 \\
& + \frac{1}{3!}\frac{\partial^3 f}{\partial u_2^3}(\mu_1,\mu_2)(u_2-\mu_2)^3 + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1^2 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2) \\
& + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^2
\end{aligned}
\tag{6.27}
$$

The residual errors of the 3rd-order Taylor series method when estimating the mean of the output can be expressed as

$$\begin{aligned} E[f(U_1,U_2)]-E[\tilde{f}(U_1,U_2)] &= \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2] \\ &+ \frac{1}{4!}\frac{\partial^4 f}{\partial u_1^4}(\mu_1,\mu_2)E[(u_1-\mu_1)^4] \\ &+ \frac{1}{4!}\frac{\partial^4 f}{\partial u_2^4}(\mu_1,\mu_2)E[(u_2-\mu_2)^4]+\ldots \end{aligned} \tag{6.28}$$

Comparing the residual errors of the 3rd-order Taylor series method in (6.28) with the residual errors of the UDR in (6.26). Both of the methods are fourth-order accurate in estimating the mean of the output, but the relative errors of the UDR comprise fewer fourth-order integration terms compared with 3rd-order Taylor series expansion. When it comes to estimating the second-order statistical moments, the relative errors of the UDR can be expressed as:

$$\begin{aligned} E[f(U_1,U_2)^2]-E[\hat{f}(U_1,U_2)^2] &= \int_{\Gamma_1}\int_{\Gamma_2}\left(f(u_1,u_2)^2-\hat{f}(u_1,u_2)^2\right)\rho(u_1)\rho(u_2)du_1du_2 \\ &= \left(\frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu_1,\mu_2)\right)^2 E[(u_1-\mu_1)^2(u_2-\mu_2)^2] \\ &+ (\frac{\partial y}{\partial u_2}(\mu_1,\mu_2)\frac{\partial^3 y}{\partial u_1^2 \partial u_2}(\mu_1,\mu_2)+ \\ &\frac{\partial y}{\partial u_1}(\mu_1,\mu_2)\frac{\partial^3 y}{\partial u_1 \partial u_2^2}(\mu_1,\mu_2))E[(u_1-\mu_1)^2(u_2-\mu_2)^2] \\ &+ \frac{1}{2!}f(\mu_1,\mu_2)\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2]+\ldots \end{aligned} \tag{6.29}$$

The residual errors for 3rd order Taylor series method can be expressed as:

$$E[f(U_1,U_2)^2]-E[\tilde{f}(U_1,U_2)^2] = \frac{1}{2!}f(\mu_1,\mu_2)\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2]+\ldots \tag{6.30}$$

Comparing the residual errors of the 3rd-order Taylor series expansion method in (6.37) with the residual errors of the UDR in (6.29). Both of the methods are also fourth-order accurate in

estimating the second-order moment of the output, but the relative errors of the UDR comprise more fourth-order integration terms compared with 3rd-order Taylor series expansion.

6.4.2 Gradient-enhanced univariate dimension reduction in 2D case

In this section, we derive the residual errors of the gradient-enhanced univariate dimension reduction (GUDR) method in estimating the mean and standard deviation for a 2D UQ problem. Consider an UQ problem that involves a bivariate function $f(u_1,u_2)$, the GUDR method approximates the original function as

$$\begin{aligned}\bar{f}(u_1,u_2) = y(u_1,\mu_2)+y(\mu_1,u_2)-y(\mu_1,\mu_2)+(u_1-\mu_1)\left(\frac{\partial f}{\partial u_1}(\mu_1,u_2)-\frac{\partial f}{\partial u_1}(\mu_1,\mu_2)\right)\\ +(u_2-\mu_2)\left(\frac{\partial f}{\partial u_2}(u_1,\mu_2)-\frac{\partial f}{\partial u_2}(\mu_1,\mu_2)\right)-\frac{\partial^2 f}{\partial u_1 u_2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2).\end{aligned} \tag{6.31}$$

The Taylor series of the involved univariate gradient terms at $(u_1=\mu_1,u_2=\mu_2)$ can be expressed by

$$\begin{aligned}\frac{\partial f}{\partial u_1}(\mu_1,u_2) = \frac{\partial f}{\partial u_1}(\mu_1,\mu_2)+\frac{\partial^2 f}{\partial u_1 u_2}(\mu_1,\mu_2)(u_2-\mu_2)+\frac{1}{2!}\frac{\partial^3 f}{\partial u_1 u_2^2}(\mu_1,\mu_2)(u_2-\mu_2)^2\\ +\frac{1}{3!}\frac{\partial^4 f}{\partial u_1 u_2^3}(\mu_1,\mu_2)(u_2-\mu_2)^3+\ldots,\end{aligned} \tag{6.32}$$

$$\begin{aligned}\frac{\partial f}{\partial u_2}(u_1,\mu_2) = \frac{\partial f}{\partial u_2}(\mu_1,\mu_2)+\frac{\partial^2 f}{\partial u_1 u_2}(\mu_1,\mu_2)(u_1-\mu_1)+\frac{1}{2!}\frac{\partial^3 f}{\partial u_1^2 u_2}(\mu_1,\mu_2)(u_1-\mu_1)^2\\ +\frac{1}{3!}\frac{\partial^4 f}{\partial u_1^3 u_2}(\mu_1,\mu_2)(u_1-\mu_1)^3+\ldots.\end{aligned} \tag{6.33}$$

The Taylor series of GUDR approximation function, $\bar{f}$ can be expressed as

$$\begin{aligned}\bar{f}(u_1,u_2) = {} & f(\mu_1,\mu_2) + \frac{\partial f}{\partial u_1}(\mu_1,\mu_2)(u_1-\mu_1) + \frac{\partial f}{\partial u_2}(\mu_1,\mu_2)(u_2-\mu_2) \\ & + \frac{1}{2!}\frac{\partial^2 f}{\partial u_1^2}(\mu_1,\mu_2)(u_1-\mu_1)^2 + \frac{1}{2!}\frac{\partial^2 f}{\partial u_2^2}(\mu_1,\mu_2)(u_2-\mu_2)^2 \\ & + \frac{\partial^2 f}{\partial u_1 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2) + \frac{1}{3!}\frac{\partial^3 f}{\partial u_1^3}(\mu_1,\mu_2)(u_1-\mu_1)^3 \\ & + \frac{1}{3!}\frac{\partial^3 f}{\partial u_2^3}(\mu_1,\mu_2)(u_2-\mu_2)^3 + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1^2 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^2(u_2-\mu_2) \\ & + \frac{1}{2!}\frac{\partial^3 f}{\partial u_1 \partial u_2^2}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^2 + \frac{1}{4!}\frac{\partial^4 f}{\partial u_1^4}(\mu_1,\mu_2)(u_1-\mu_1)^4 \\ & + \frac{1}{4!}\frac{\partial^4 f}{\partial u_2^4}(\mu_1,\mu_2)(u_2-\mu_2)^4 + \frac{1}{3!}\frac{\partial^4 f}{\partial u_1^3 \partial u_2}(\mu_1,\mu_2)(u_1-\mu_1)^3(u_2-\mu_2) \\ & + \frac{1}{3!}\frac{\partial^4 f}{\partial u_1 \partial u_2^3}(\mu_1,\mu_2)(u_1-\mu_1)(u_2-\mu_2)^3 + \ldots\end{aligned} \tag{6.34}$$

If we compare the Taylor series expansion of the original function with the GUDR approximation, all of the Taylor series terms in (6.34) are contained in (6.22), and the residual errors can be expressed as

$$f(u_1,u_2) - \bar{f}(u_1,u_2) = \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(0,0)u_1^2 u_2^2 + \ldots \tag{6.35}$$

which only include fourth and higher-order terms. When we estimate the first-order statistical moment of the output, the GUDR approximation generates the same results as the UDR as

$$E[f(u_1,u_2)] - E[\bar{f}(u_1,u_2)] = \frac{1}{2!2!}\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2] + \ldots \tag{6.36}$$

When we estimate the second-order statistical moment of the output, the residual errors for GUDR approximation can be expressed as

$$E[f(u_1,u_2)^2] - E[\bar{f}(u_1,u_2)^2] = \frac{1}{2!}f(\mu_1,\mu_2)\frac{\partial^4 f}{\partial u_1^2 \partial u_2^2}(\mu_1,\mu_2)E[(u_1-\mu_1)^2(u_2-\mu_2)^2] + \ldots. \tag{6.37}$$

Compared to the residual errors of the 3rd-order Taylor series method, shown in (6.37), the residual errors of the GUDR approximation contain the same fourth-order integration terms.

6.4.3 Example code to implement computational graph transformation in GUDR

In this section, we provide the example code to implement the computational graph transformation method in GUDR for a 2D UQ problem. The Python code below assumes we have the univariate model and gradient functions' evaluations on quadrature points of each input dimension, along with the function and gradient evaluations on the mean of the input. The code shows how to use NumPy's Einstein summation function to expand the sizes of certain inputs so that we can generate the correct tensor-grid evaluations for the GUDR approximation function.

```
import numpy as np
k # number of quadrature points in each dimension
u_1 # quadrature points in u_1, size: (k,1)
u_2  # quadrature points in u_2, size: (k,1)
f1_u1 #  quadrature points evaluations of f_1(u_1), size (k,1)
f2_u2 #  quadrature points evaluations of f_2(u_2), size (k,1)
df1_du # quadrature points evaluations of df_1/du(u_1), size (k,2)
df2_du # quadrature points evaluations of df_2/du(u_2), size (k,2)
mean_u # mean of the inputs, size(2,)
f_mean_u = # function evaluation on mean of the inputs, size (1,)
df_du_mean_u = # gradient evaluation on mean of the inputs, size (2,)
d2f_du2_mean_u = # Hessian evaluation on mean of the inputs, size (2,2)
total_points = k**2 # total input points k**2

# Use Einsum and reshape operations to expand the size of certain inputs
u_1 = np.einsum('i...,p...->ip...', u_1 , np.ones(k))
u_1 = np.reshape(u_1 , (total_points,))
u_2 = np.einsum('p...,i...->ip...', u_2 , np.ones(k))
u_2 = np.reshape(u_2 , (total_points,))
```

```
f1_u1 = np.einsum('i...,p...->ip...', f1_u1, np.ones(k))
f1_u1 = np.reshape(f1_u1, (total_points,))
f2_u2 = np.einsum('p...,i...->ip...', f2_u2, np.ones(k))
f2_u2 = np.reshape(f2_u2, (total_points,))
df1_du = np.einsum('i...,p...->ip...', df1_du, np.ones(k))
df1_du = np.reshape(df1_du, (total_points, 2))
df2_du = np.einsum('p...,i...->ip...', df2_du, np.ones(k))
df2_du = np.reshape(df2_du, (total_points, 2))

# Evaluate GUDR approximation function on tensor-grid quadrature points
f_gudr = np.zeros(total_points,)
for i in range(total_points):
  u =  np.array([u_1[i], u_2[i]])
  a1 = np.array([0, u_2[i]- mean_u[1]])
  a2 = np.array([u_1[i]- mean_u[0], 0])
  term1 = f1_u1[i] + f2_u2[i] - f_mean_u
  term2 = np.dot(a1, df1_du[i,:]) + np.dot(a2, df2_du[i,:])- np.dot((u-
                                    mean_u), df_du_mean_u)
  term3 = -0.5*np.einsum('...i,...i->...', (u-mean_u).T.dot(
                                    d2f_du2_mean_u - np.diag(np.diag(
                                    d2f_du2_mean_u))), (u-mean_u))
  f_gudr[i] = term1 + term2+ term3

# f_gudr: GUDR approximation function evaluations on tensor-grid
                                  quadrature points, size (k^2,)
```

Acknowledgments

The material presented in this paper is, in part, based upon work supported by DARPA under grant No. D23AP00028-00.

Chapter 6, is taken, in part, from the material [102] which has been submitted for

publication as Wang, B., Orndorff, N. C., Sperry, M., and Hwang, J.T., "A gradient-enhanced univariate dimension reduction method for uncertainty propagation," Aerospace Science and Technology. The dissertation author was the primary investigator and author of this paper.

Chapter 7

Graph-accelerated large-scale multidisciplinary design optimization under uncertainty of a laser-beam-powered aircraft

This chapter presents a detailed case study for a laser-beam-powered aircraft design problem. The focus is on applying the graph-accelerated NIPC methods to solve a large-scale multidisciplinary design optimization under uncertainty (MDOUU) problem. This study aims to not only compare the results of MDO and MDOUU but also highlight the effectiveness of the graph-accelerated NIPC method in tackling MDOUU problems.

The advancements in power-beaming technology have opened up possibilities for the development of unmanned aircraft primarily powered by laser beams. This breakthrough offers the potential to reduce the reliance on onboard energy storage, prompting the exploration of new aircraft designs that can fully leverage this unique property. Expanding upon previous research in MDO for similar aircraft, as discussed in [72], we develop a practical OUU problem tailored to a specific mission scenario. This problem formulation accounts for three pivotal random variables linked to flight conditions. The numerical results offer a comparative analysis of the designs optimized through MDO and MDOUU for specific metrics. The findings highlight that the MDOUU-optimized design exhibits greater robustness and reliability when subjected to the defined uncertain inputs. Furthermore, we demonstrate a substantial acceleration in optimization time, achieving a fivefold improvement by implementing the computational graph transformation

method, AMTC. This efficiency enhancement results in a computational cost for solving the MDOUU problem that is merely double that of solving the MDO problem.

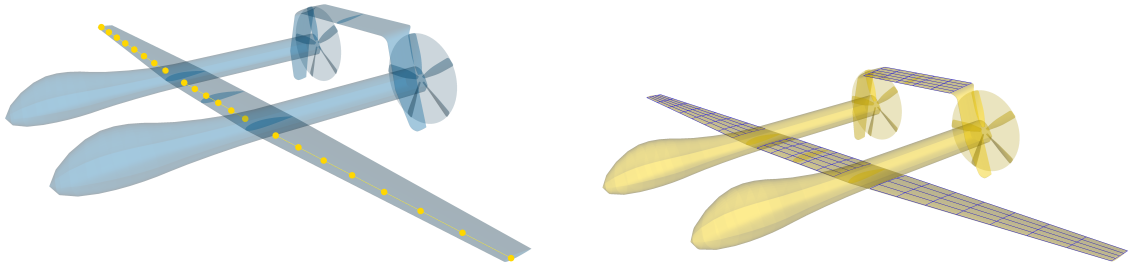
7.1 Problem setup

In this problem, we aim to optimize the conceptual design of a twin-fuselage, high-altitude unmanned airplane. This aircraft is uniquely engineered to accommodate a central pod that receives power from an incoming laser beam and converts it to electrical power via an internal photoelectric array. The airplane concept is shown in Fig. 3.6. We consider a simple mission scenario where the aircraft is in cruise condition while being powered only by a laser beam emitted from a ground station. The laser engagement scenario is shown in Fig. 6.8. The detailed descriptions for all of the sub-models for each discipline can be found in [72].

7.1.1 Computational model and optimization formulations

Our computational model involves multiple disciplines, including aerodynamics, structural analysis, thermal dissipation, power beaming, and stability assessment. The aerodynamic solver employs the vortex-lattice method to calculate the aerodynamic forces generated by the wing and tail. In the structural analysis component, we assume a wing-box structure and determine the Von-Mises stress at specific evaluation points on the beam cross-section, focusing on areas where bending and shear stress are expected to be the greatest. For the power-beaming model, we leverage data generated from the High Energy Laser End-to-End Operational Simulation (HELEEOS) [23] software to compute beam propagation loss. The meshes used for the structure and aerodynamics solvers are both shown in Fig. 7.1 Additionally, Fig. 7.2 provides a visual representation of the wing box structure.

We begin by establishing an MDO problem, as outlined in [72]. The objective here is to find the optimal aircraft design, with a focus on minimizing the total weight of the aircraft while ensuring compliance with specified constraints. A detailed formulation of the MDO problem is presented in Tab. 7.1, and the associated XDSM diagram is depicted in Fig. 7.3. It is important



(a) 1D beam meshes for the structure solver

(b) 2D aerodynamics meshes for the aerodynamics solver

Figure 7.1. Meshes for structure and aerodynamics solvers

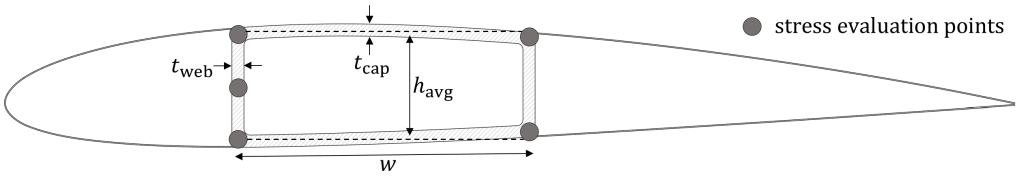


Figure 7.2. The wing box structure with stree evaluation points

to note that, in addition to the optimization constraints outlined in Tab. 7.1, several requirements are inherently embedded within the computational model. These encompass considerations such as endurance, heat dissipation, and static margin, all of which contribute to shaping the final aircraft design.

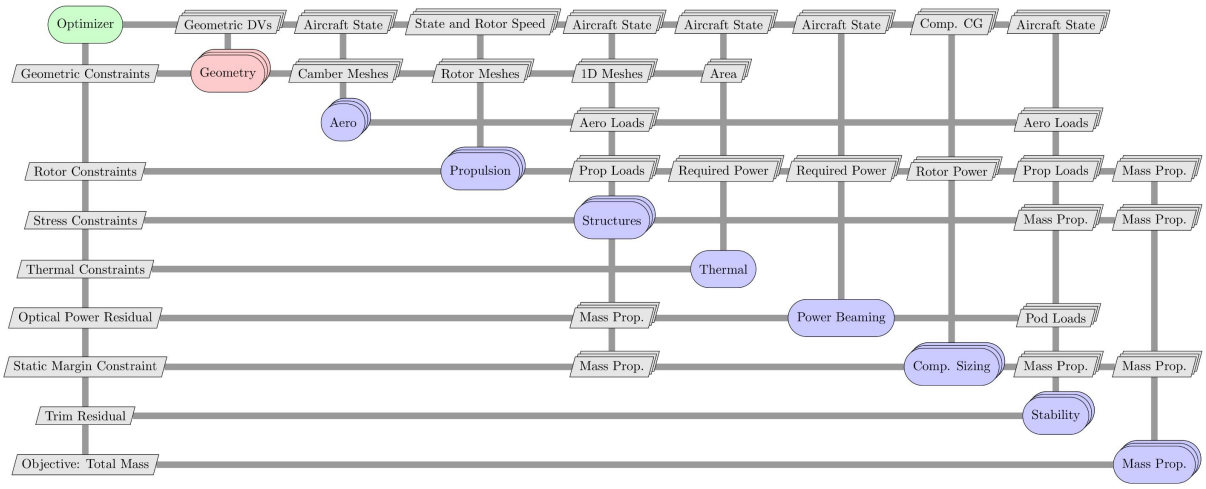


Figure 7.3. XDSM diagram of the computational mdoel

Upon examination of the MDO formulation presented in Tab. 7.1, we identified three

Table 7.1. MDO problem formulation

	DESCRIPTION	RANGE (UNITS)	QUANTITY
OBJECTIVE	Aircraft total weight	w (kg)	1
DESIGN VARIABLES	Wing span scaling	$-10 \leq b_w \leq 10$	1
	Tail span scaling	$-4 \leq b_t \leq 4$	1
	Tail incidence	$-10 \leq i_{ht} \leq 10$ (deg)	1
	Pitch angle	$-10 \leq \theta \leq 10$ (deg)	1
	Mach number	$0.1 \leq m \leq 0.3$	1
	Rotor speed	$500 \leq n \leq 3000$(rpm)	1
	Aperture diameter	$0.25 \leq d_a \leq 1$ (m)	1
	Wing beam cap thickness	$0.001 \leq t_{\text{cap}} \leq 0.01$(m)	14
	Wing beam web thickness	$0.001 \leq t_{\text{web}} \leq 0.01$ (m)	14
			Total: 35
CONSTRAINTS	Optical power residual	$r_{\text{o}} \geq 0$	1
	Trim residual	$r(\vec{x}) \leq 1e^{-6}$	1
	Maximum stress	$s \leq s_{\max}$	70
			Total: 72

pivotal uncertain inputs: flight altitude, horizontal distance, and payload weight. These random variables are pivotal in this problem, as the aircraft operates across diverse mission scenarios. To account for the variability of these three uncertain inputs within our optimization framework, we formulate the MDOUU problem, as outlined in Tab. 7.2. This formulation falls into the category of robust design optimization, wherein the objective function and inequality constraints are expressed as linear combinations of the statistical moments of the stochastic outputs.

7.1.2 Graph-accelerated NIPC for MDOUU

Using the graph-accelerated NIPC methods to solve MDOUU problems involves a four-step process:

1. Define the computational model in graph-based modelling software and generate its computational graph

2. Generate the tensor-grid quadrature rule based on computational graph analysis.

3. Generate the modified computational graph using AMTC for efficient tensor-grid evaluations.

Table 7.2. MDOUU problem formulation

	DESCRIPTION	RANGE (UNITS)	QUANTITY
OBJECTIVE	Aircraft total weight	$\mu_W + 3\sigma_W$ (kg)	1
DESIGN VARIABLES	Wing span scaling	$-10 \leq b_w \leq 10$	1
	Tail span scaling	$-4 \leq b_t \leq 4$	1
	Tail incidence	$-10 \leq i_{ht} \leq 10$ (deg)	1
	Pitch angle	$-10 \leq \theta \leq 10$ (deg)	1
	Mach number	$0.1 \leq m \leq 0.3$	1
	Rotor speed	$500 \leq n \leq 3000$(rpm)	1
	Aperture diameter	$0.25 \leq d_a \leq 1$ (m)	1
	Wing beam cap thickness	$0.001 \leq t_{\text{cap}} \leq 0.01$(m)	14
	Wing beam web thickness	$0.001 \leq t_{\text{web}} \leq 0.01$ (m)	14
			Total: 35
RANDOM VARIABLES	Altitude	$\mathscr{U}(4000, 6000)$ (m)	1
	Horizontal distance	$\mathscr{U}(220, 280)$ (km)	1
	Payload weight	$\mathscr{U}(40, 60)$ (kg)	1
			Total: 3
CONSTRAINTS	Optical power residual	$\mu_{R_o} - 3\sigma_{R_o} \geq 0$	1
	Trim residual	$\mu_R + 3\sigma_R \leq 1e^{-6}$	1
	Maximum stress	$\mu_S + 3\sigma_S \leq S_{\max}$	70
			Total: 72

4. Solve the MDOUU problem in the double-loop framework using the modified computational graph for evaluations.

In the MDOUU context, the UQ problems at different optimization iterations involve the same computational graph and uncertain inputs but with different design variable values. Consequently, there's no need to regenerate the modified computational graph for each optimization iteration. Instead, we only need to update the design variable values on the modified computational graph at each optimization iteration. As a result, using the graph-accelerated NIPC approach to solve the MDOUU problem barely adds up any computational cost, and the acceleration it brought is directly related to the reduced repeated evaluations on the operation level.

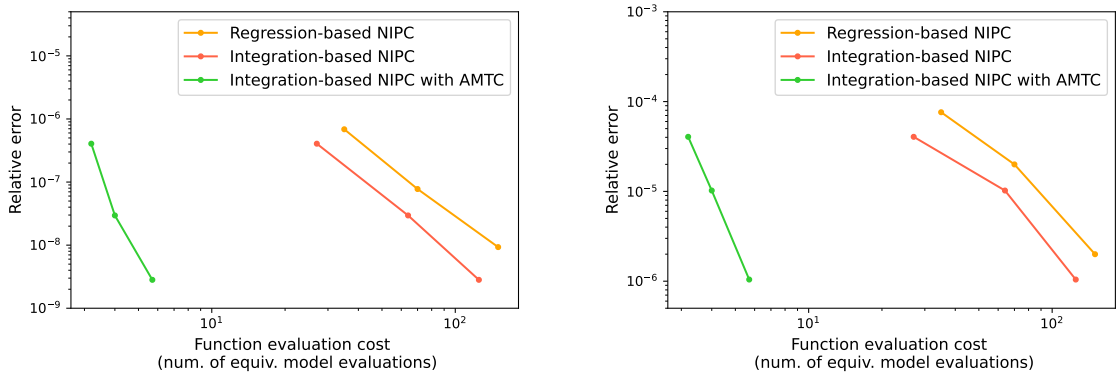
7.2 Numerical results

In this study, we first investigate the performance of the graph-accelerated NIPC method by applying it to the solution of the UQ problem with the initial design of the aircraft. We then compare its performance with integration-based NIPC and regression-based NIPC methods. The UQ results are also used to decide on the number of quadrature points we use in the MDOUU problem. Following this, we tackle an MDO problem, as outlined in Tab.7.1, aimed at refining the aircraft's initial design. After optimizing the design through MDO, this improved design serves as the starting point for addressing the MDOUU problem, detailed in Tab.7.2. We then draw a comparison between the results of the MDO and MDOUU, evaluating them across various metrics. This comparison underscores the effectiveness of the AMTC method. To ensure an equitable comparison in terms of computational time, we opt for the finite-difference approach to calculate gradients for all involved optimization problems.

7.2.1 Comparison of the UQ methods

We assess the performance of the graph-accelerated NIPC method by conducting a comparative analysis against two other UQ techniques. The objective here is to compute the mean and standard deviation of the objective function for the initial design of the aircraft, under the effect of the three uncertain inputs we defined. The three methods under evaluation are full-grid integration-based NIPC (integration-based NIPC), full-grid integration-based NIPC with AMTC (integration-based NIPC with AMTC) and regression-based NIPC. The regression-based NIPC results are generated using the software package Chaospy [22], with a random sampling approach.

The convergence plots of the UQ methods are shown in Fig. 7.4. The results here show that by using the AMTC method, the model evaluation cost of the integration-based NIPC method can be reduced by more than one order of magnitude making it the clear winner compared to the other two implemented methods. The significant acceleration here can be explained by



(a) Mean of the output **(b)** Standard deviation of the output

Figure 7.4. Convergence plots of the UQ methods

Table 7.3. Sparsity ratio of the uncertain inputs

Uncertain inputs	Sparsity ratio
Altitude	98.3%
Horizontal distance	0.4%
Payload weight	3.2%

looking at the sparsity ratio of each uncertain input shown in Tab. 7.3. The sparsity ratio was first introduced in [100] as a measure for the percentage of dependent computational cost for each uncertain input. From it, we observe that, out of the three uncertain inputs, two of them have only a small percentage of the computational cost dependent on them. This means, that most of the computationally dominant operations in this model are only dependent on one uncertain input, and only need to be evaluated on the k quadrature points in that dimension. By eliminating the redundant evaluations on the operation level, AMTC significantly accelerates the model evaluations on the tensor-grid.

7.2.2 Comparison of the MDO and MDOUU results

Based on the UQ results, we choose to use the integration-based NIPC to solve the UQ sub-problem in MDOUU. The full-grid Gauss quadrature points with 3 nodes in each dimension are used to ensure a balance between accuracy and efficiency, and the AMTC method is used to accelerate the model evaluations on tensor-grid quadrature points.

Table 7.4. Comparison of MDO and MDOUU results

	Initial design	MDO-optimized design	MDOUU-optimized design
Design variables:			
b_w	0	0.1145	0.1886
b_t	0	0.3149	0.3313
i_{ht}	0	1.674	1.573
θ	0	-0.3495	-0.5340
m	0.15	0.1	0.1
n	1500	1083	1082
d_a	0.75	0.5318	0.5368
MDO objective	1024	588	607
MDO constraints:			
Violation No.	19	0	0
Average violation (%)	$> 100\%$	-	-
OUU objective	1123	634	655
OUU constraints:			
Violation No.	20	4	0
Average violation (%)	$> 100\%$	1.2%	-
Number of opt. iterations	-	58	26
Optimization time	-	38 min	1h 18 min (with AMTC) 7h 52 min (without AMTC)

The MDO and MDOUU-optimized designs are compared with the initial design in Tab. 7.4, and the visualization of MDO and MDOUU-optimized designs can be found in Fig. 7.5. The results reveal a notable distinction between the MDO-optimized design and the initial design. It is noteworthy that the MDO problem required 38 minutes for the optimizer to converge, leading to a significant enhancement over the initial design. Specifically, it resulted in a 40% reduction in the MDO objective function while satisfying all MDO constraints.

However, upon evaluating the MDOUU constraint functions on the MDO-optimized design, we discovered that four constraints were violated, with an average violation of approximately 1.2%. This implies that the MDO-optimized design may not fully comply with certain constraint functions when subjected to changes in flight conditions. Consequently, by employing MDOUU to enhance the MDO-optimized design, we achieved an optimal design that successfully meets all of the MDOUU constraint functions. While the OUU-optimized design exhibits a higher objective value compared to the MDO-optimized design, the fact that it satisfies all

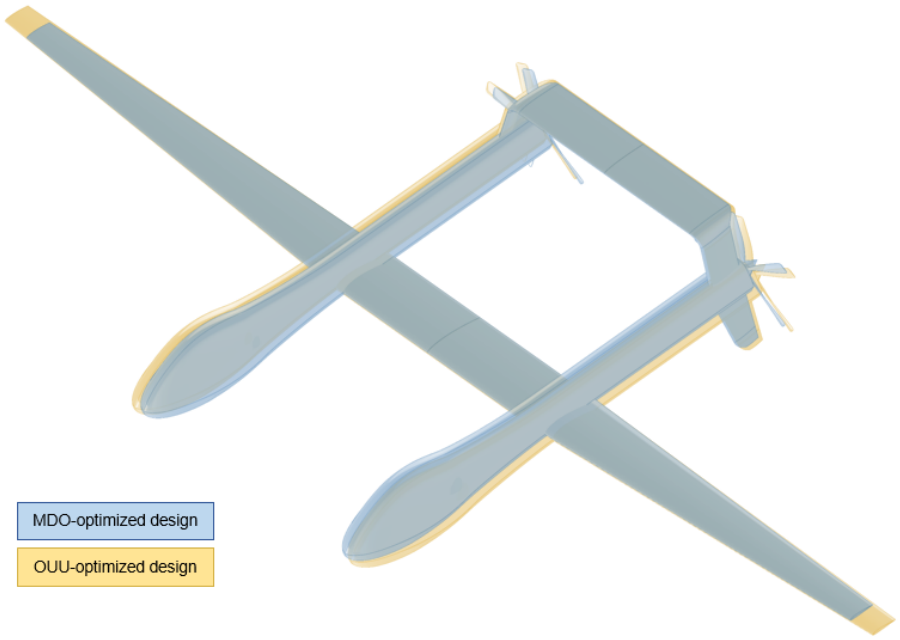


Figure 7.5. Visualizations of the MDO and MDOUU results

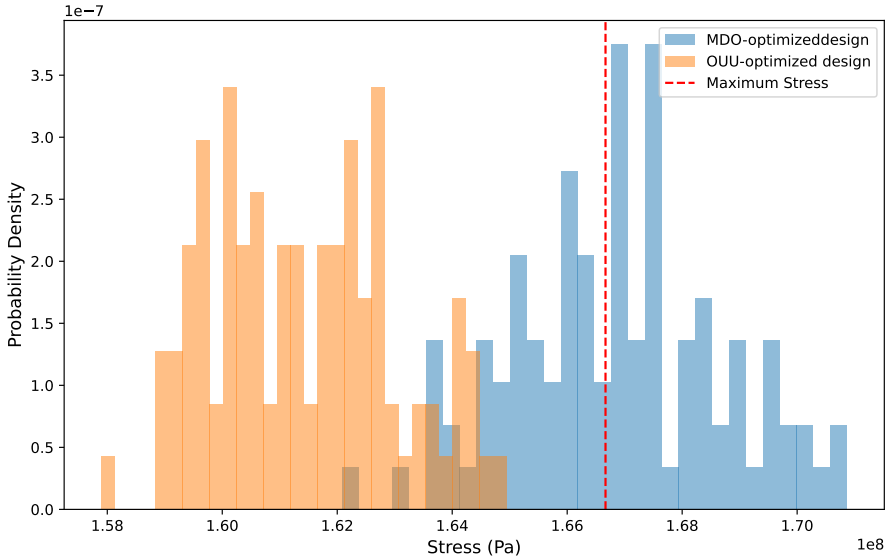


Figure 7.6. Comparison of the probability density distributions of a stress constraint function output

MDOUU constraints underscores its resilience when operating under varying flight conditions, characterized as uncertain inputs in the MDOUU problem. The comparison between MDO and

MDOUU results under the effect of the uncertain inputs in one of the stress constraint functions that the MDO-optimized design violated is shown in Fig. 7.6. The results show that almost half of the probability density distribution for the MDO-optimized design is over the maximum stress limit while the entire distribution of the MDOUU-optimized design is below the maximum stress limit. Furthermore, in terms of computational efficiency in solving the MDOUU problem, the AMTC method plays a crucial role by reducing the total optimization time by approximately fivefold. Consequently, the optimization time for solving the MDOUU problem is merely twice that of solving the MDO problem

From an aircraft design perspective, it is worth noting that the MDOUU-optimized design exhibits distinct characteristics compared to the MDO-optimized counterpart. Specifically, the MDOUU-optimized design features a larger wing span, a greater tail span, and a generally thicker beam cross-section. These design differences can be attributed to the inherent conservatism of the MDOUU approach. The larger wing area in the MDOUU-optimized design is strategically designed to generate sufficient lift, ensuring the aircraft can carry potentially heavier payloads. Additionally, the stronger wing structure is designed to guarantee compliance with constraint requirements under a wide range of flight conditions. In summary, the MDOUU-optimized design prioritizes robustness and reliability across varying scenarios, which leads to these specific design modifications.

Acknowledgments

The material presented in this paper is, in part, based upon work supported by DARPA under grant No. D23AP00028-00

Chapter 7, in part, is a reprint of the material [99] as it appears in Wang, B., Orndorff, N. C., Joshy, A. J., and Hwang, J. T., "Graph-accelerated large-scale multidisciplinary design optimization under uncertainty of a laser-beam-powered aircraft," AIAA SciTech 2024 Forum, 2024, p. 0169. The dissertation author was the primary investigator and author of this paper.

Chapter 8

Conclusion

The objective of this thesis is to devise efficient efficient uncertainty propagation methods that can be seamlessly integrated into the existing gradient-based MDO framework. In this chapter, I summarize the contributions made by my research and offer recommendations for future work.

8.1 Summary of contributions

My first contribution is the development of a computational graph transformation method named *Accelerated Model Evaluations on Tensor Grids using Computational Graph Transformation* (AMTC). This method enhances the efficiency of tensor-grid evaluations by leveraging the computational graph sparsity inherent in the model. By integrating AMTC with the non-intrusive polynomial chaos (NIPC) method, it effectively addresses a wide range of low-dimensional uncertainty quantification (UQ) problems, typically involving fewer than four uncertain inputs. This combined approach, known as the graph-accelerated NIPC method, also opens up a new avenue of research to design the quadrature rule in a desired tensor structure so that the AMTC method can make the best use of the computational graph sparsity.

My second contribution is the development of a framework for designing partially tensor-structured quadrature rules compatible with the AMTC method. This framework constructs the quadrature rules using a tailored quadrature method and selects the optimal tensor structure

by analyzing the model's computational graph. This advancement extends the applicability of the graph-accelerated NIPC methods to a broader range of UQ problems, accommodating higher-dimensional random spaces, typically with fewer than ten uncertain inputs.

My third contribution is the development of AS-AMTC, a method that integrates graph-accelerated NIPC methods with the active subspace (AS) dimension reduction technique to address high-dimensional UQ problems. During the creation of AS-AMTC, I also developed AS-NIPC, which combines the AS method with integration-based NIPC methods to solve high-dimensional UQ problems. Building on AS-NIPC, AS-AMTC merges the AS method with graph-accelerated NIPC methods, providing a more efficient solution for high-dimensional UQ problems (with more than ten uncertain inputs) with multidisciplinary systems.

My fourth contribution is the development of a UQ method called *gradient-enhanced univariate dimension reduction* (GUDR). This method improves the accuracy of the well-known univariate dimension reduction (UDR) method in estimating second and higher-order statistical moments. The GUDR method incorporates univariate gradient terms into the UDR approximation function and utilizes the AMTC for efficient tensor-grid evaluations. By employing an efficient automatic differentiation technique, this method maintains the linear scalability of the UDR method while significantly enhancing accuracy in estimating higher-order statistical moments.

8.2 Recommendations for future work

This thesis presents a suite of graph-accelerated UQ methods designed for tackling multidisciplinary UQ and MDOUU problems across various scenarios. Despite their potential, significant work remains to fully realize these methods' capabilities in addressing practical, large-scale MDOUU problems. Below, I outline several key questions that should be addressed in future research:

First, a critical step towards fully integrating graph-accelerated NIPC methods into the gradient-based MDO framework is the combination of the AMTC method with automatic

differentiation to achieve efficient analytical derivative computations using modified computational graphs. Gradient-based optimization methods utilizing analytical derivatives can achieve better-than-linear scalability, which is essential for enabling large-scale MDO. By integrating AMTC with automatic differentiation, graph-accelerated NIPC methods can significantly improve the efficiency of both model and gradient evaluations, thereby expanding the applicability of large-scale MDOUU.

Second, the AS-AMTC and AS-NIPC methods proposed in this thesis are currently applicable only to single-output computational models. To extend these methods to address MDOUU problems, it is essential to develop strategies for managing computational models with multiple outputs, such as those involving objective function outputs and constraint function outputs in MDOUU problems.

Third, in this thesis, the graph-accelerated NIPC methods have been utilized solely for estimating the statistical moments of the QoIs. The quadrature rules employed are designed from the perspective of polynomial exactness, making them effective for this purpose. However, to expand these methods for solving reliability-based design optimization problems, it is necessary to develop more efficient quadrature methods capable of estimating complex risk measures, such as the probability of failure or conditional value at risk.

Bibliography

[1] Martín Abadi. Tensorflow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN international conference on functional programming*, pages 1–1, 2016.

[2] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[3] Galib Abumeri and Christos Chamis. Non-deterministic multidisciplinary optimization of engine structures. In *8th Symposium on Multidisciplinary Analysis and Optimization*, page 4926, 2000.

[4] Eytan J Adler and Joaquim RRA Martins. Hydrogen-powered aircraft: Fundamental concepts, key technologies, and environmental impacts. *Progress in Aerospace Sciences*, 141:100922, 2023.

[5] Ivo Babuška, Fabio Nobile, and Raúl Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

[6] Mathieu Balesdent, Nicolas Bérend, Philippe Dépincé, and Abdelhamid Chriette. A survey of multidisciplinary design optimization methods in launch vehicle design. *Structural and Multidisciplinary optimization*, 45(5):619–642, 2012.

[7] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018.

[8] Garo Bedonian, Jason E Hicken, and Edwin Forster. A gradient-enhanced partition-of-unity surrogate model and adaptive sampling strategy for non-analytic functions. *Structural and Multidisciplinary Optimization*, 66(7):167, 2023.

[9] Einat Neumann Ben-Ari and David M Steinberg. Modeling data from computer experiments: an empirical comparison of kriging with mars and projection pursuit regression. *Quality Engineering*, 19(4):327–338, 2007.

[10] Géraud Blatman and Bruno Sudret. Sparse polynomial chaos expansions and adaptive stochastic finite elements using a regression approach. *Comptes rendus mécanique*, 336(6):518–523, 2008.

[11] Komahan Boopathy, Markus P Rumpfkeil, and Raymond M Kolonay. Robust optimization of a wing under structural and material uncertainties. In *17th AIAA Non-Deterministic Approaches Conference*, page 0920, 2015.

[12] Mohamed Amine Bouhlel, John T. Hwang, Nathalie Bartoli, Rémi Lafage, Joseph Morlier, and Joaquim R. R. A. Martins. A python surrogate modeling framework with derivatives. *Advances in Engineering Software*, page 102662, 2019.

[13] Benjamin J Brelje and Joaquim RRA Martins. Electric, hybrid, and turboelectric fixed-wing aircraft: A review of concepts, models, and design approaches. *Progress in Aerospace Sciences*, 104:1–19, 2019.

[14] Loic Brevault, Mathieu Balesdent, and Ali Hebbal. Multi-objective multidisciplinary design optimization approach for partially reusable launch vehicle design. *Journal of Spacecraft and Rockets*, 57(2):373–390, 2020.

[15] Loïc Brevault, Mathieu Balesdent, Jérôme Morio, et al. *Aerospace system analysis and optimization in uncertainty*. Springer, 2020.

[16] Bruce Christianson. Automatic hessians by reverse accumulation. *IMA Journal of Numerical Analysis*, 12(2):135–150, 1992.

[17] Hyonho Chun and Sündüz Keleş. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.

[18] Paul G Constantine, Eric Dow, and Qiqi Wang. Active subspace methods in theory and practice: applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4):A1500–A1524, 2014.

[19] George B Dantzig. Linear programming under uncertainty. *Management science*, 1(3-4):197–206, 1955.

[20] Thomas D Economon, Francisco Palacios, Sean R Copeland, Trent W Lukaczyk, and Juan J Alonso. Su2: An open-source suite for multiphysics simulation and design. *Aiaa Journal*, 54(3):828–846, 2016.

[21] Hans A Eschenauer and Niels Olhoff. Topology optimization of continuum structures: a review. *Appl. Mech. Rev.*, 54(4):331–390, 2001.

[22] Jonathan Feinberg and Hans Petter Langtangen. Chaospy: An open source tool for designing methods of uncertainty quantification. *Journal of Computational Science*, 11:46–57, 2015.

[23] Steven T Fiorino, Robb M Randall, Richard J Bartell, John D Haiducek, Mark F Spencer, and Salvatore J Cusumano. Field measurements and comparisons to simulations of high energy laser propagation and off-axis scatter. In *Free-Space Laser Communications X*, volume 7814, pages 186–196. SPIE, 2010.

[24] KB Fragkos, EM Papoutsis-Kiachagias, and KC Giannakoglou. pfosm: An efficient algorithm for aerodynamic robust design based on continuous adjoint and matrix-vector products. *Computers & Fluids*, 181:57–66, 2019.

[25] Rudolf J Freund. The introduction of risk into a programming model. *Econometrica: Journal of the econometric society*, pages 253–263, 1956.

[26] Victor Gandarillas, Anugrah Jo Joshy, Mark Z. Sperry, Alexander K. Ivanov, and John T Hwang. A graph-based methodology for constructing computational models that automates adjoint-based sensitivity analysis. *Structural and Multidisciplinary Optimization*, 2024.

[27] Thomas Gerstner and Michael Griebel. Numerical integration using sparse grids. *Numerical algorithms*, 18(3-4):209–232, 1998.

[28] SF Ghoreishi and DL Allaire. Adaptive uncertainty propagation for coupled multidisciplinary systems. *AIAA journal*, 55(11):3940–3950, 2017.

[29] Philip E Gill, Walter Murray, and Michael A Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131, 2005.

[30] Andrew Glaws and Paul G Constantine. Gaussian quadrature and polynomial approximation for one-dimensional ridge functions. *SIAM Journal on Scientific Computing*, 41(5):S106–S128, 2019.

[31] Gene H Golub and John H Welsch. Calculation of gauss quadrature rules. *Mathematics of computation*, 23(106):221–230, 1969.

[32] Justin S Gray, John T Hwang, Joaquim RRA Martins, Kenneth T Moore, and Bret A Naylor. Openmdao: An open-source framework for multidisciplinary design, analysis, and optimization. *Structural and Multidisciplinary Optimization*, 59(4):1075–1104, 2019.

[33] Clyde Gumbert, Perry Newman, and Gene Hou. Effect of random geometric uncertainty on the computational design of a 3d flexible wing. In *20th AIAA applied aerodynamics conference*, page 2806, 2002.

[34] Raphael T Haftka and Ramana V Grandhi. Structural shape optimization—a survey. *Computer methods in applied mechanics and engineering*, 57(1):91–106, 1986.

[35] Wanxin He, Gang Li, Changting Zhong, and Yixuan Wang. A novel data-driven sparse polynomial chaos expansion for high-dimensional problems based on active subspace and sparse bayesian learning. *Structural and Multidisciplinary Optimization*, 66(1):29, 2023.

[36] Jason E Hicken. Inexact hessian-vector products in reduced-space differential-equation constrained optimization. *Optimization and Engineering*, 15:575–608, 2014.

[37] Serhat Hosder, Robert Walters, and Rafael Perez. A non-intrusive polynomial chaos method for uncertainty propagation in cfd simulations. In *44th AIAA aerospace sciences meeting and exhibit*, page 891, 2006.

[38] Zhen Hu and Sankaran Mahadevan. A single-loop kriging surrogate modeling for time-dependent reliability analysis. *Journal of Mechanical Design*, 138(6):061406, 2016.

[39] Zhen Hu, Sankaran Mahadevan, and Dan Ao. Uncertainty aggregation and reduction in structure–material performance prediction. *Computational Mechanics*, 61(1):237–257, 2018.

[40] John T Hwang, Dae Young Lee, James W Cutler, and Joaquim RRA Martins. Large-scale multidisciplinary optimization of a small satellite's design and operation. *Journal of Spacecraft and Rockets*, 51(5):1648–1663, 2014.

[41] John T Hwang and Joaquim RRA Martins. A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives. *ACM Transactions on Mathematical Software (TOMS)*, 44(4):1–39, 2018.

[42] L Jaeger, Christian Gogu, Stéphane Segonds, and Christian Bes. Aircraft multidisciplinary design optimization under both model and design variables uncertainty. *Journal of Aircraft*, 50(2):528–538, 2013.

[43] Antony Jameson. Aerodynamic design via control theory. *Journal of scientific computing*, 3:233–260, 1988.

[44] Lucas Janson, Edward Schmerling, and Marco Pavone. Monte carlo motion planning for robot trajectory optimization under uncertainty. In *Robotics Research: Volume 2*, pages 343–361. Springer, 2017.

[45] Cyrus Jilla and David Miller. A multiobjective, multidisciplinary design optimization methodology for the conceptual design of distributed satellite systems. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, page 5491, 2002.

[46] Brandon A Jones, Alireza Doostan, and George H Born. Nonlinear propagation of orbit uncertainty using non-intrusive polynomial chaos. *Journal of Guidance, Control, and Dynamics*, 36(2):430–444, 2013.

[47] Anugrah Jo Joshy, Ryan Dunn, Mark Sperry, Victor E Gandarillas, and John T Hwang. An sqp algorithm based on a hybrid architecture for accelerating optimization of large-scale systems. In *AIAA AVIATION 2023 Forum*, page 4263, 2023.

[48] Anugrah Jo Joshy and John T Hwang. Unifying monolithic architectures for large-scale system design optimization. *AIAA Journal*, 59(6):1953–1963, 2021.

[49] Susan Joslyn and Sonia Savelli. Communicating forecast uncertainty: Public perception of weather forecast uncertainty. *Meteorological Applications*, 17(2):180–195, 2010.

[50] Arthur B Kahn. Topological sorting of large networks. *Communications of the ACM*, 5(11):558–562, 1962.

[51] Shugo Kaneko and Joaquim RRA Martins. Fleet design optimization of package delivery unmanned aerial vehicles considering operations. *Journal of Aircraft*, 60(4):1061–1077, 2023.

[52] Irfan Kaymaz. Application of kriging method to structural reliability problems. *Structural safety*, 27(2):133–151, 2005.

[53] Gaetan Kenway and Joaquim RRA Martins. Aerostructural shape optimization of wind turbine blades considering site-specific winds. In *12th AIAA/ISSMO multidisciplinary analysis and optimization conference*, page 6025, 2008.

[54] Vahid Keshavarzzadeh, Felipe Fernandez, and Daniel A Tortorelli. Topology optimization under uncertainty via non-intrusive polynomial chaos expansion. *Computer Methods in Applied Mechanics and Engineering*, 318:120–147, 2017.

[55] Vahid Keshavarzzadeh, Robert M Kirby, and Akil Narayan. Numerical integration in multiple dimensions with designed quadrature. *SIAM Journal on Scientific Computing*, 40(4):A2033–A2061, 2018.

[56] Isaac I Kim, Bruce McArthur, and Eric J Korevaar. Comparison of laser beam propagation at 785 nm and 1550 nm in fog and haze for optical wireless communications. In *Optical wireless communications III*, volume 4214, pages 26–37. Spie, 2001.

[57] Dongjin Lee and Sharif Rahman. Practical uncertainty quantification analysis involving statistically dependent random variables. *Applied Mathematical Modelling*, 84:324–356, 2020.

[58] Daejin Lim, Hyeongseok Kim, and Kwanjung Yee. Uncertainty propagation in flight performance of multirotor with parametric and model uncertainties. *Aerospace Science and Technology*, 122:107398, 2022.

[59] Carlos López, Aitor Baldomir, and Santiago Hernández. Deterministic versus reliability-based topology optimization of aeronautical structures. *Structural and Multidisciplinary Optimization*, 53:907–921, 2016.

[60] Jiaqi Luo and Feng Liu. Statistical evaluation of performance impact of manufacturing variability by an adjoint method. *Aerospace Science and Technology*, 77:471–484, 2018.

[61] Jiaqi Luo, Yao Zheng, and Feng Liu. Optimal tolerance allocation in blade manufacturing by sensitivity-based performance impact evaluation. *Journal of Propulsion and Power*, 36(4):632–638, 2020.

[62] Nora Luthen, Stefano Marelli, and Bruno Sudret. Sparse polynomial chaos expansions: Literature survey and benchmark. *SIAM/ASA Journal on Uncertainty Quantification*, 9(2):593–649, 2021.

[63] Charles A Mader, Joaquim RRA Martins, Juan J Alonso, and Edwin Van Der Weide. Adjoint: An approach for the rapid development of discrete adjoint solvers. *AIAA journal*, 46(4):863–873, 2008.

[64] Joaquim RRA Martins, Juan J Alonso, and James J Reuther. High-fidelity aerostructural design optimization of a supersonic business jet. *Journal of Aircraft*, 41(3):523–530, 2004.

[65] Joaquim RRA Martins and Andrew B Lambe. Multidisciplinary design optimization: a survey of architectures. *AIAA journal*, 51(9):2049–2075, 2013.

[66] Joaquim RRA Martins, Peter Sturdza, and Juan J Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software (TOMS)*, 29(3):245–262, 2003.

[67] Jérôme Morio. Global and local sensitivity analysis methods for a physical system. *European journal of physics*, 32(6):1577, 2011.

[68] Leo WT Ng and Karen E Willcox. Monte carlo information-reuse approach to aircraft conceptual design optimization under uncertainty. *Journal of Aircraft*, 53(2):427–438, 2016.

[69] Andrew Ning and K Dykes. Understanding the benefits and limitations of increasing maximum rotor tip speed for utility-scale wind turbines. In *Journal of physics: conference series*, volume 524, page 012087. IOP Publishing, 2014.

[70] Fabio Nobile, Raúl Tempone, and Clayton G Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.

[71] Nicholas C Orndorff, Darshan Sarojini, Luca Scotzniovsky, Hyunjune Gill, Seongkyu Lee, Zeyu Cheng, Shuofeng Zhao, Chris Mi, and John T Hwang. Air-taxi transition trajectory optimization with physics-based models. In *AIAA SCITECH 2023 Forum*, page 0324, 2023.

[72] Nicholas C Orndorff, Bingran Wang, Marius L Ruh, Andrew Fletcher, and John T Hwang. Gradient-based sizing optimization of power-beaming-enabled aircraft. In *AIAA AVIATION 2023 Forum*, page 4019, 2023.

[73] Dhanesh Padmanabhan. *Reliability-based optimization for multidisciplinary system design*. University of Notre Dame, 2003.

[74] F. Pappenberger, K. J. Beven, N. M. Hunter, P. D. Bates, B. T. Gouweleeuw, J. Thielen, and A. P. J. de Roo. Cascading model uncertainty from medium range weather forecasts (10 days) through a rainfall-runoff model to flood inundation predictions within the european flood forecasting system (effs). *Hydrology and Earth System Sciences*, 9(4):381–393, 2005.

[75] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Optimal model management for multifidelity monte carlo estimation. *SIAM Journal on Scientific Computing*, 38(5):A3163–A3194, 2016.

[76] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *Siam Review*, 60(3):550–591, 2018.

[77] Olivier Pironneau. On optimum design in fluid mechanics. *Journal of fluid mechanics*, 64(1):97–110, 1974.

[78] Sharif Rahman and Heqin Xu. A univariate dimension-reduction method for multi-dimensional integration in stochastic mechanics. *Probabilistic Engineering Mechanics*, 19(4):393–408, 2004.

[79] Daniel Raymer. *Aircraft design: a conceptual approach*. American Institute of Aeronautics and Astronautics, Inc., 2012.

[80] Marius L Ruh, Andrew Fletcher, Darshan Sarojini, Mark Sperry, Jiayao Yan, Luca Scotzniovsky, Sebastiaan P van Schie, Michael Warner, Nicholas C Orndorff, Ru Xiang, et al. Large-scale multidisciplinary design optimization of a nasa air taxi concept using a comprehensive physics-based system model. In *AIAA SCITECH 2024 Forum*, page 0771, 2024.

[81] Marius L Ruh and John T Hwang. Fast and robust computation of optimal rotor designs using blade element momentum theory. *AIAA Journal*, 61(9):4096–4111, 2023.

[82] Markus P Rumpfkeil. Optimizations under uncertainty using gradients, hessians, and surrogate models. *AIAA journal*, 51(2):444–451, 2013.

[83] Nikolaos V Sahinidis. Optimization under uncertainty: state-of-the-art and opportunities. *Computers & chemical engineering*, 28(6-7):971–983, 2004.

[84] Darshan Sarojini, Marius L Ruh, Anugrah Jo Joshy, Jiayao Yan, Alexander K Ivanov, Luca Scotzniovsky, Andrew H Fletcher, Nicholas C Orndorff, Mark Sperry, Victor E Gandarillas, et al. Large-scale multidisciplinary design optimization of an evtol aircraft using comprehensive analysis. In *AIAA SciTech 2023 Forum*, page 0146, 2023.

[85] Claudia Schillings, Stephan Schmidt, and Volker Schulz. Efficient shape optimization for certain and uncertain aerodynamic design. *Computers & Fluids*, 46(1):78–87, 2011.

[86] Lucien A Schmit. Structural design by systematic synthesis. In *Proceedings of the Second National Conference on Electronic Computation, ASCE, Sept., 1960*, 1960.

[87] Lucien A Schmit and William A Thornton. *Synthesis of an airfoil at supersonic Mach number*. National Aeronautics and Space Administration, 1965.

[88] Sabet Seraj and Joaquim R Martins. Aerodynamic shape optimization of a supersonic transport considering low-speed stability. In *AIAA Scitech 2022 Forum*, page 2177, 2022.

[89] Christopher Silva, Wayne R Johnson, Eduardo Solis, Michael D Patterson, and Kevin R Antcliff. Vtol urban air mobility concept vehicles for technology development. In *2018 Aviation Technology, Integration, and Operations Conference*, page 3847, 2018.

[90] Sergei Abramovich Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk*, volume 148, pages 1042–1045. Russian Academy of Sciences, 1963.

[91] Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.

[92] Mark Sperry, Kavish Kondap, and John T Hwang. Automatic adjoint sensitivity analysis of models for large-scale multidisciplinary design optimization. In *AIAA AVIATION 2023 Forum*, page 3721, 2023.

[93] Armin Tabandeh, Gaofeng Jia, and Paolo Gardoni. A review and assessment of importance sampling methods for reliability analysis. *Structural Safety*, 97:102216, 2022.

[94] Mishal Thapa, Sameer B Mulani, and Robert W Walters. Adaptive weighted least-squares polynomial chaos expansion with basis adaptivity and sequential adaptive sampling. *Computer Methods in Applied Mechanics and Engineering*, 360:112759, 2020.

[95] Hua-Ping Wan, Zhu Mao, Michael D Todd, and Wei-Xin Ren. Analytical uncertainty quantification for modal frequencies with structural parameter uncertainty using a gaussian process metamodel. *Engineering Structures*, 75:577–589, 2014.

[96] Bingran Wang, Anugrah Jo Joshy, and John T Hwang. Equality-constrained engineering design optimization using a novel inexact quasi-newton method. *AIAA Journal*, 60(11):6157–6167, 2022.

[97] Bingran Wang, Nicholas C Orndorff, and John T Hwang. Optimally tensor-structured quadrature rule for uncertainty quantification. In *AIAA SCITECH 2023 Forum*, page 0741, 2023.

[98] Bingran Wang, Nicholas C Orndorff, and John T Hwang. Graph-accelerated non-intrusive polynomial chaos expansion using partially tensor-structured quadrature rules. *arXiv preprint arXiv:2403.15614*, 2024.

[99] Bingran Wang, Nicholas C Orndorff, Anugrah J Joshy, and John T Hwang. Graph-accelerated large-scale multidisciplinary design optimization under uncertainty of a laser-beam-powered aircraft. In *AIAA SCITECH 2024 Forum*, page 0169, 2024.

[100] Bingran Wang, Nicholas C Orndorff, Mark Sperry, and John T Hwang. High-dimensional uncertainty quantification using graph-accelerated non-intrusive polynomial chaos and active subspace methods. In *AIAA AVIATION 2023 Forum*, page 4264, 2023.

[101] Bingran Wang, Nicholas C Orndorff, Mark Sperry, and John T Hwang. Extension of graph-accelerated non-intrusive polynomial chaos to high-dimensional uncertainty quantification through the active subspace method. *arXiv preprint arXiv:2405.05556*, 2024.

[102] Bingran Wang, Nicholas C Orndorff, Mark Sperry, and John T Hwang. A gradient-enhanced univariate dimension reduction method for uncertainty propagation. *arXiv preprint arXiv:2403.15622*, 2024.

[103] Bingran Wang, Mark Sperry, Victor E Gandarillas, and John T Hwang. Efficient uncertainty propagation through computational graph modification and automatic code generation. In *AIAA AVIATION 2022 Forum*, page 3997, 2022.

[104] Bingran Wang, Mark Sperry, Victor E Gandarillas, and John T Hwang. Accelerating model evaluations in uncertainty propagation on tensor grids using computational graph transformations. *Aerospace Science and Technology*, 145:108843, 2024.

[105] Norbert Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936, 1938.

[106] Jeffrey M Wooldridge. Applications of generalized method of moments estimation. *Journal of Economic perspectives*, 15(4):87–100, 2001.

[107] Dongbin Xiu and Jan S Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139, 2005.

[108] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2):619–644, 2002.

[109] Wen Yao, Xiaoqian Chen, Wencai Luo, Michel Van Tooren, and Jian Guo. Review of uncertainty-based multidisciplinary design optimization methods for aerospace vehicles. *Progress in Aerospace Sciences*, 47(6):450–479, 2011.

[110] Thomas A Zang. *Needs and opportunities for uncertainty-based multidisciplinary design methods for aerospace vehicles*. National Aeronautics and Space Administration, Langley Research Center, 2002.