A flexible uncertainty quantification method for linearly coupled multi-physics systems

Xiao Chen*, Brenda Ng, Yunwei Sun, Charles Tong*

Lawrence Livermore National Laboratory, 7000 East Ave., Livermore, California 94550, USA

Abstract

This paper presents a novel approach to building an integrated uncertainty quantification (UQ) methodology suitable for modern-day component-based approach for multi-physics simulation development. Our “hybrid” UQ methodology supports independent development of the most suitable UQ method, intrusive or non-intrusive, for each physics module by providing an algorithmic framework to couple these “stochastic” modules for propagating “global” uncertainties. We address algorithmic and computational issues associated with the construction of this hybrid framework. We demonstrate the utility of such a framework on a practical application involving a linearly coupled multi-species reactive transport model.

Keywords: uncertainty quantification; polynomial chaos; stochastic modeling; reactive transport; operator splitting

1. Introduction

The use of modeling and simulation in scientific and engineering exploration has been ubiquitous in recent years due to rapid advances in numerical methods and the wide availability of high performance computers. Accurate prediction of engineering systems using simulation tools, however, has not been well-established because of the presence of various sources of uncertainty, such as model parameter uncertainties, model form uncertainties, experimental data uncertainties, etc. Therefore, to enhance the credibility and confidence in predictive simulations, rigorous quantification of uncertainties is critical. Thus, uncertainty quantification (UQ) has emerged as an indispensable component in developing robust models and simulations for science and engineering.

UQ encompasses a suite of analysis tasks such as forward propagation of uncertainties, sensitivity analysis, surrogate or response surface modeling, model validation/calibration, and risk assessment. The execution of these tasks (which include treatment of system uncertainty) may incur orders of magnitude more computational demand than the same tasks without taking uncertainty into account. For complex simulation models such as those in many multi-physics and multi-scale applications, performing credible UQ analysis continues to be computationally prohibitive. The search for efficient methods for propagation of uncertainty has thus become a primary thrust in UQ research.

*Corresponding authors
Email addresses: chen7301@llnl.gov (Xiao Chen), tong10@llnl.gov (Charles Tong)
Historically, there have been two tracks of method development for propagating uncertainties: non-intrusive and intrusive methods. For non-intrusive methods, uncertainties are introduced through input parameters to a selected simulation model in the form of probability distributions. Sample points, drawn from these probability distributions, are propagated through the deterministic simulation model. The resulting outputs are used to learn the input-output relationships and to compute model statistics. Variants of non-intrusive methods differ in their sampling schemes, examples of which include Latin hypercube [1], Quasi-Monte Carlo sampling [2], and importance sampling [3, 4].

For real-world multi-physics applications, non-intrusive methods have become quite popular due to simplicity in implementation – interfacing sampling methods with deterministic simulation codes requires minimal effort in most cases. However, this approach, which treats the simulation codes as black boxes, obscures the detailed evolution of uncertainties within the simulation models, offers limited insights about the underlying stochastic dynamics beyond the ensemble statistics from the input-output samples.

To facilitate the understanding of stochastic dynamics, one might turn to intrusive methods that require knowledge of the governing equations that underlie the simulation codes. Intrusive methods explicitly represent input uncertainties via a parametric representation (e.g., Neuman series, Taylor series, or orthogonal polynomials [5, 6, 7, 8, 9]). The input uncertainties are propagated through the governing equations, and the output uncertainties are explicitly derived (as a mathematical expression bound by a parametric form depending on the intrusive method chosen) as opposed to being derived from sample statistics. By construction, the output uncertainties constitute the “solution” to the stochastic version of the governing equations, so this approach provides insights about the impact of uncertainties as they are propagated through the simulation model. This “transparency” is at the cost of required access to the governing equations, as well as increased development time because intrusive methods need to be embedded within the simulation codes.

To keep pace with the ever-increasing complexity of modern multi-physics models, it is clear that neither purely intrusive or purely non-intrusive frameworks would be adequate. Purely intrusive schemes would require complete rewrite of all components (or modules), which may not be feasible since some modules may come from commercial or open source software packages. Sampling-based schemes, on the other hand, consider a multi-physics simulation as a monolithic application without naturally distinguishing the coupling between modules and the evolution of input uncertainties through various modules, thus potentially overlooking dependencies and eventual simplifications. Instead of representing a stochastic multi-physics model entirely in terms of polynomial chaos or through sampling of the associated deterministic model, or trying to combine the two together, we advocate a different kind of hybridization, in the modular sense.

Our objective is to develop a computational framework and the associated algorithms for propagating uncertainties and global sensitivities in a multi-physics/multi-component environment where some modules can benefit from using intrusive UQ methods while other modules may not afford any intrusive UQ treatments. Our approach is based on the concept of plug-and-play, the component-based development practice that manages model complexity and facilitates rapid software development through “clean” separation between individual physics components. The advantage of this approach is that each module can have its own UQ method – be it purely intrusive or non-intrusive (and even a combination of both) – and all these methods can work together synergistically in a weakly or strongly coupled manner. Thus, the resulting UQ framework is hybrid on the module level as each module can still be intrusive or non-intrusive. This approach facilitate the handling of increasing model complexity through providing flexibility in continuous enhancement of individual modules.
physics components.

Our technical contribution is the design and implementation of such a “modularly hybrid” UQ framework, which required overcoming algorithmic and computational aspects that are needed to achieve “global” uncertainty propagation within the context of modular UQ computations. To successfully develop this methodology, we addressed the problem of how to accurately propagate uncertainties between individual physics modules with minimal loss of uncertainty information.

To demonstrate our framework, we solve a transient reactive transport problem that assumes an unknown but constant flow. This type of problem is traditionally solved by operator splitting (i.e., solving the transport and reaction modules in sequence at each time step), so there is a natural decomposition into modules that is well-suited to test our framework. We show that different modules can be solved using different UQ methods while still maintaining global uncertainty propagation throughout the system. Potential benefits of our modularly hybrid UQ framework include: (1) higher degree of flexibility in selecting UQ schemes for each physics module; and (2) the ability to naturally track evolution of uncertainties in detail (at grid point level).

Our framework can naturally handle nonlinearly coupled modules and modules with spatial randomness. For clarity, we focus our discussion on linearly coupled modules as it provides the most intuitive setting for understanding the fundamental concepts behind our proposed framework. Once the fundamental concepts are laid down, the framework’s applicability to more realistic models is straightforward through extending to higher dimensions or propagating extra information between modules.

The outline of the paper is as follows: In Section 2, we provide a brief review of uncertainty quantification methods, such as intrusive methods (based on polynomial chaos) and non-intrusive methods (based on sampling). In Section 3, we explain the general theory behind our modularly hybrid UQ framework, with primary focus on linearly coupled modules, followed by brief discussions about more realistic modules, such as those that are nonlinearly coupled or contains spatial randomness. In Section 4, we introduce the multi-species reactive transport problem and describe the steps to apply our framework to solve the problem. Lastly, we discuss results in Section 5 and conclude in Section 6.

2. Uncertainty quantification methods

In this section, we briefly review intrusive and non-intrusive uncertainty quantification methods.

2.1. Intrusive methods: polynomial chaos

One popular intrusive method is the stochastic Galerkin method using polynomial chaos expansion (PCE) [5, 6], which has been successfully applied to UQ analysis in many applications, including solid mechanics [5], transport in heterogeneous media [7], combustion [10], and fluid mechanics [8, 9]. The method represents the solution of stochastic equations using PCE. The stochastic equations are rewritten in terms of these solution expansions then projected onto an appropriate set of basis functions to derive a new, larger set of deterministic equations, from which the PCE coefficients are solved then used to reconstruct the output uncertainties.

Formally, polynomial chaos is a member of the set of homogeneous chaos, which was first defined by Wiener [11] as the span of Hermite polynomial functionals of a Gaussian random process. It was later pioneered by Ghanem and Spanos [5] for quantifying uncertainties in various applications. Subsequently, Xiu and Karniadakis [9] generalized Wiener’s idea of chaos for various classes
(based on Askey’s classification [12]) of orthogonal polynomials that are coupled to their associated stochastic processes.

Consider a probability space \((\Theta, \Sigma, P)\) where \(\Theta\) is the sample space, \(\Sigma\) is a \(\sigma\)-algebra on \(\Theta\) (non-empty collection of subset of \(\Omega\) that is closed under complementation and countable unions of its member), and \(P\) is a probability measure (mapping \(\Sigma\) to \([0, 1]\)). Let’s consider \(\mathcal{R}\)-valued random variables \(X\) defined on \((\Theta, \Sigma, P)\):

\[
X : \Theta \mapsto \mathcal{R}
\]

and denote by \(L_2(\Theta, P)\) the set of second-order random variables, such that for \(\forall X \in L_2(\Theta, P)\), we have \(E[|X|^2] < \infty\) where the mean \(E\) is defined as:

\[
E[X] = \int_{\Theta} X(\theta) \, dP(\theta).
\]

The set of second-order random variables form a Hilbert space with respect to the inner product:

\[
\langle X, Y \rangle = E[XY] = \int_{\Theta} X(\theta)Y(\theta) \, dP(\theta) = \int_{\mathcal{R}} \int_{\mathcal{R}} XY \, P(X, Y) \, dXdY
\]

where \(P(X, Y)\) is the joint probability density function of \(X\) and \(Y\). The associated norm of \(X\) is \(\sqrt{E[|X|^2]}\).

An important concept in representing uncertainty using polynomial chaos is that one can express a second-order random variable \(X\) that only involves \(m\) random variables \(\xi_1, \cdots, \xi_m\) as a sum of orthogonal polynomials:

\[
X = X_0 + \sum_{j=1}^{m} X_{j_1} \Psi_1(\xi_{j_1}(\theta)) + \sum_{j_1=1}^{m} \sum_{j_2=1}^{j_1} X_{j_1, j_2} \Psi_2(\xi_{j_1}(\theta), \xi_{j_2}(\theta)) + \cdots
\]

where \(\Psi_k(\xi_{j_1}, \cdots, \xi_{j_k})\) are polynomial chaos of order \(k\) in the variables \((\xi_{j_1}, \cdots, \xi_{j_k})\) with its type and domain depending on the distributions imposed on the \(m\)-dimensional random variables \(\xi = \{\xi_1, \cdots, \xi_m\}\).

To simplify notation, this multi-dimensional expansion is usually mapped term-by-term to a single index form given by:

\[
\hat{X} = \sum_{j=0}^{\kappa} X_j \Psi_j(\xi)
\]

where \(\Psi_j(\xi)\)’s are polynomial chaoses in single-index form and \((\kappa + 1)\) is the total number of terms used for polynomial order \(\leq p\) such that:

\[
\kappa + 1 = \frac{(p + m)!}{p!m!}.
\]

All the polynomial chaoses are mutually orthogonal with regards to the inner product associated to the space spanned by the random variables \(\xi\). In particular, when uniform distribution is assumed for the \(m\) random variables, the associated orthogonal polynomials are the Legendre polynomials which have the following property:

\[
\int_{-1}^{1} L_j(x) L_i(x) \, dx = \frac{2}{2j + 1} \delta_{j, i}.
\]
where $\delta_{jl}$ is the Kronecker delta. The Legendre polynomials satisfy the recurrence relation:

$$(l + 1) L_{l+1} (x) - (2l + 1) x L_l (x) + l L_{l-1} (x) = 0$$

with $L_0(x) = 1$ and $L_1(x) = x$. Using PCE, it is straightforward to verify that the approximate mean of $X$ (using $(\kappa + 1)$ terms) is $X_0$ and the approximate variance is:

$$\int_{-1}^{1} \left[ \sum_{j=0}^{\kappa} X_j \psi_j (\xi) - x_0 \psi_0 (\xi) \right]^2 d\xi = \sum_{j=1}^{\kappa} \left[ X_j^2 \langle \psi_j^2 \rangle \right].$$

(9)

Furthermore, let $\hat{\xi}$ be a subset of $\xi$ and $\hat{\Psi}$ be the set of all Legendre polynomials of order up to $p$ involving only the random variables in $\hat{\xi}$. Then the partial variance (Sobol’ index) for this subset can be calculated by:

$$S (\hat{\xi}) \approx \frac{\sum_{\psi \in \hat{\Psi}} \left[ X_j^2 \langle \psi_j^2 \rangle \right]}{\sum_{j=1}^{\kappa} \left[ X_j^2 \langle \psi_j^2 \rangle \right]}.$$  

(10)

To model uncertainty in a given partial differential equation system with parametric uncertainties, the independent and the dependent random variables are both represented by the appropriate PCE. A Galerkin projection is then performed onto each of the $(\kappa + 1)$ orthogonal polynomials giving rise to a coupled system of $(\kappa + 1)$ equations, which are discretized and solved for the coefficients $X_j$’s. Finally, the coefficients are substituted back into the expansion.

Intrusive methods generally, when the input-output relationship is sufficiently smooth, has the potential to be more computationally efficient than non-intrusive methods [13, 14]. However, this is at the expense of (1) higher development effort, since implementation require major modifications to existing deterministic codes, and (2) additional overhead in solving the larger set of equations, which might be structurally different enough to warrant new solvers. These two challenges might be too cumbersome and time-consuming to overcome for most complex multi-physics multi-scale models. As a result, intrusive methods have not been as widely adopted as non-intrusive methods because their advances have not been keeping pace with the fast increases in the complexities of modern multi-physics models.

2.2. Non-intrusive methods

Non-intrusive methods consist of generating samples of random variables based on their distributions via a fixed sampling scheme. Different sampling schemes generate samples with different “space coverage” properties. The samples are then propagated through the model by running the model repeatedly with the sample inputs. The outputs of interest are collected and the desired statistics such as mean and standard deviation are computed. Again, let $m$ be the number of random variables and let $P(\xi_1, \xi_2, \ldots, \xi_m)$ be the corresponding joint probability distribution function. Let $S \in \mathbb{R}^{N \times m}$ be the set of $N$ samples drawn from this distribution where each row of $S$ is a single sample of the input random variables. Let $Y \in \mathbb{R}^N$ be the (univariate) outputs corresponding to the $N$ “input” samples. Then, the sample approximation of the mean and standard deviation is:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} Y_i$$

(11)

$$\sigma = \frac{1}{N - 1} \sqrt{\sum_{i=1}^{N} (Y_i - \mu)^2}.$$  

(12)
While straightforward to implement, the non-intrusive approach suffers from poor computational efficiency: the convergence rate of the computed mean is only \( O(1/\sqrt{N}) \) for the Monte Carlo sampling method. Thus, to increase the accuracy of the computed mean by one decimal would require increasing \( N \) by 100-fold. Despite the fact that other sampling strategies (e.g., Latin hypercube [1], Quasi-Monte Carlo [2], importance sampling [3, 4]) have been proposed to improve convergence, existing sampling methods are still inadequate to handle large-scale models.

To mitigate this problem of slow convergence, one might exploit the fact that, in many real-life models, the outputs of interest are smooth function of the input random variables. This motivates the use of response surface models (also known as surrogate models, meta-models, and emulators) that interpolate the values of the output variables. Interpolation is generally computationally much cheaper than the actual simulation runs, thus allowing for quicker and larger-scale generation of samples. The choice of interpolation scheme is heavily dependent on the specific application problem.

It should be noted that the non-intrusive form of the polynomial chaos method (sometimes called spectral collocation) is analogous to a special type of polynomial regression scheme. An advantage of this scheme is that if the output of interest is a polynomial function of the input random variables, the convergence rate of computing the basic statistics is exponential with respect to the sample size, and the basic statistical measures are readily available from the polynomial chaos coefficients associated with the output variables.

Once a good response surface model has been constructed, Sobol’ indices can be computed in several ways such as Sobol’ method [15], McKay’s method [16], or direct numerical integration. For example, McKay’s method makes use of the replicated Latin hypercube (r-LHS) sampling strategy. In the r-LHS design, each random variable \( \xi_i, i = 1, ..., m \), takes on distinct values \( x_{ij}, j = 1, ..., L \), where \( L \) is called the number of levels. These values are to be randomized and re-used \( R \) times to form the final design which has \( N = LR \) sample points.

The estimator for the Sobol’ index corresponding to \( \xi_i \) is given by:

\[
S(\xi_i) = \frac{1}{V(Y)} \left\{ \frac{1}{L} \sum_{j=1}^{L} [\bar{Y}(\xi_i = x_{ij}) - \bar{Y}]^2 - \frac{1}{LR^2} \sum_{j=1}^{L} \sum_{r=1}^{R} [Y^{(r)}(\xi_i = x_{ij}) - \bar{Y}(\xi_i = x_{ij})]^2 \right\}
\]

where \( \bar{Y}(\xi_i = x_{ij}) = \frac{1}{R} \sum_{r=1}^{R} Y^{(r)}(\xi_i = x_{ij}) \) and \( Y^{(r)}(\xi_i = x_{ij}) \) is the output of the sample point corresponding to \( \xi_i = x_{ij} \) in the \( r^{th} \) replication; \( \bar{Y} \) and \( V(Y) \) are the aggregate mean and variance of \( Y \), respectively.

3. Modularly hybrid UQ framework: theory

Hybrid methods seek to bridge the gap between the practicality of non-intrusive methods and the potential efficiency and robustness of intrusive methods. The main idea is to integrate both methods into a UQ framework of a complex multi-physics model.

The concept of hybridization for UQ has been realized in many forms ([17, 18, 19]). For example, Hosder et al. [20] presented a non-intrusive polynomial chaos method, which uses samples to construct the equations for solving the PCE coefficients. Surana and Banaszuk [21] proposed probabilistic waveform relaxation, applicable for networked systems with weakly coupled subsystems, which integrates generalized polynomial chaos and probabilistic collocation into waveform relaxation. Abgrall et al. [22] proposed a semi-intrusive method that solves the deterministic problem over an extended domain that encapsulates uncertainty, which requires less modification to the
deterministic solver and can also handle more general types of input distributions. Constantine et al. [19, 23] combined the intrusive Galerkin method with non-intrusive stochastic allocation and demonstrated their method on a one-way decoupled system. Guadagnini et al. [24] employed random domain decomposition to divide physical domain for composite material according to a large-scale distribution of geological units and then quantify the local-level uncertainty within each disjoint, statistically homogeneous unit. Our work is most similar to [19], but allows uncertainty introduced in one module to influence the stochastic dependent variables in the second module, making it applicable to more general multi-physics models.

Most of the aforementioned methods are only applicable for specialized problems that satisfy the methods’ assumptions of decoupled or weakly-coupled physics modules. In contrast, our take on “hybrid UQ” is much more general. Our notion of “hybrid” refers to techniques that facilitate the use of different intrusive and non-intrusive methods for strongly-coupled modules in a multi-physics simulation. This is a largely unexplored concept with many possible research directions. For example, one can use an intrusive method in one single-physics module and a non-intrusive method in another module, and then piece their uncertainties together (as done in this paper). Alternatively, a single-physics module can be made hybrid with some of its parameters handled intrusively and others handled non-intrusively to handle nonlinear dependencies.

In this section, we introduce the general theory behind our modularly hybrid UQ framework for global uncertainty propagation. This UQ framework is different from other types of hybrid frameworks: while other hybrid frameworks combine properties of intrusive and non-intrusive methods into one monolithic algorithm for uncertainty propagation, our framework is hybrid in the modular sense, in that different algorithms (pure or combined versions of intrusive or non-intrusive) can be applied to different modules in the multi-physics system. Our modularly hybrid framework achieves this through a plug-and-play design, which facilitates fast and easy module development and maintenance. This design also makes it possible to exploit the structure in complex multi-physics equations to facilitate inference in a divide-and-conquer manner whenever possible.

3.1. Module operators

We begin by explaining the operators that handle information between the modules. For illustrative purposes, we will explain these operators within the context of forward uncertainty propagation for linearly coupled modules.

Let $K$ be the number of independent physics modules in the multi-physics system. Each module has embedded UQ, meaning each module has a “stochastic” (intrusive or non-intrusive) solver that propagates uncertainty from inputs to outputs at each time step. Let $\xi = \{\xi_1, \xi_2, \ldots, \xi_K\}$ be $K$ disjoint subsets of independent second-order random variables, where $\xi_k$ denotes module $k$’s internal random variables. Also, let $u(x, \xi)$ be the unknown global quantities of interest for the full multi-physics system. At each time step, each module $k$ contributes local information to update the global quantities of interest as follows:

$$u^*(x, \xi) = M_k(\xi_k, u(x, \xi))$$

(14)

where $M_k$ is a stochastic solver for module $k$ that takes as inputs $\xi_k$ and $u(x, \xi)$, and solves for $u^*(x, \xi)$. Note that $u^*(x, \xi)$ is only a partial update with respect to the full multi-physics model. After all the modules have traversed in a pre-defined order, then the uncertainty update for a given time step is completed.

To make our discussion concrete, we explain our framework within the context of two linearly coupled modules. Using operator-splitting, we assume that the full multi-physics model has been
decomposed into two modules that work in concert to update the state from time $t$ to $t + 1$:

$$u^{t+1/2}(x, \xi) = M_1(\xi_1, u^t(x, \xi))$$

$$u^{t+1}(x, \xi) = M_2(\xi_2, u^{t+1/2}(x, \xi))$$

(15)

where $M_1$ and $M_2$ are the respective solvers for modules 1 and 2, and $\xi_1$ and $\xi_2$ are their respective internal random variables such that $\xi = \{\xi_1, \xi_2\}$. Eq. (15) reflects the status quo where the UQ treatment is not modularly hybrid. Since $u^{t+1/2}(x, \xi)$, the partial solution from $M_1$, is fed directly into $M_2$, changes introduced in $M_1$ will need to be reflected in $M_2$. This is because both $M_1$ and $M_2$ require knowledge of the global stochastic space $\xi$. For example, if one adds a new variable to $M_1$, this new variable will be part of $\xi$, and thus $M_2$ will need to be modified to handle the new $\xi$ that has been augmented with the new variable.

We now introduce a new framework, the *modularly hybrid framework*, where changes in $M_1$ will not impact the $M_2$’s implementation (and vice versa). The key is in making each module “self-inclusive” via the use of inter-module operators that translates each module’s local information into a common format that can be understood by all modules. Let’s assume that $M_1$ has a PCE-based solver and $M_2$ has a sampling-based solver. Furthermore, let’s consider a PCE representation as the common format for the solution, so each module’s partial solution will be expressed as:

$$u^*(x, \xi) = \sum_{i=0}^\kappa u_i(x)\psi_i(\xi)$$

(16)

where $(\kappa + 1)$ is the number of PCE coefficients as defined in Eq. (6).

Our algorithmic goal is to develop techniques that require only the local uncertainties ($\xi_k$) to be propagated in the modules. (This is in contrast to Eq. (14) which shows that the global uncertainties ($\xi$) must be propagated even in module $k$.) Towards this end, we introduce the restriction and prolongation operators. The restriction operator maps information from the global uncertain parameter space $\xi$ to the module’s local parameter space $\xi_k$. The prolongation operator does the inverse mapping from the module’s local parameter space $\xi_k$ back to the global uncertain parameter space $\xi$. As such, these operators are specific to the UQ method embedded in a given module. For linearly coupled modules, we can, without any loss of information, decompose each module’s computation into independent subproblems that can be solved in parallel. In other words, we can write:

$$u^*(x, \xi) = \sum_{i=1}^{n_k} P_{ki} \tilde{M}_{ki}(\xi_k, R_{ki} u(x, \xi))$$

(17)

where $R_{ki}$ and $P_{ki}$ are restriction and prolongation operators for module $k$’s subproblem $i$, and $n_k$ is the number of subproblems. We use the solver $M_{ki}$ to ensure only local uncertainties are propagated within module $k$.

For PCE-based modules, the implementation of $R_{ki}$ and $P_{ki}$ depends on whether the modules are coupled linearly or nonlinearly. Recall that $p$ is the PCE polynomial order, $m$ is the number of random variables in $\xi$, and $m_k$ is the number of internal random variables in $\xi_k$. For the linear case, the number of independent subproblems will be $n_k = (m - m_k + p)!/(m - m_k)!p!$ (proof given in the next subsection). In each of these subproblems, $R_{ki}$ and $P_{ki}$ are analogous to the “scatter” and “gather” operations respectively. For the nonlinear case, the construction of $R_{ki}$ and $P_{ki}$ becomes more algorithmically involved. A feasible approach would be to use sampling in conjunction with PCE, in particular, apply sampling to the external variables so that the PCE representations of the
nonlinear coefficients for each sample point are expressed only in terms of the internal variables to the module at hand. The nonlinear case will be further outlined in 3.3.

For sampling-based modules, the treatment is the same for both linear and nonlinear cases. In this setting, $M_{kl}$ is simply a deterministic solver for a fixed sample $\xi$ and $n_k$ is naturally the sample size. $R_{kl}$ maps $u(x, \xi)$ onto a single point in the random parameter space, and $P_{kl}$ transforms the results back to polynomial coefficients in the PC space.

Developing this hybrid UQ framework requires a software infrastructure that provides building blocks, such as the $R_{kl}$’s and $P_{kl}$’s for different scenarios, and thus permit reuse for different applications. The objective of this framework design is to enable independent module development while preserving the ability to propagate global uncertainty. We have implemented a prototype of this framework.

3.2. Decomposition property for linear equations

As alluded to previously, for the linear case, it is possible to decompose a PCE-based module’s computation into parallel, independent subproblems. Here, we present a theoretical property for linearly coupled equations that validates this decomposition. This property is significant in that it demonstrates that our “separability” of uncertainty treatment preserves the “global” nature of uncertainty propagation. Thus, this property legitimizes the “plug-and-play”-motivated approach to uncertainty propagation, that is, a code developer for a PCE-based module need not be concerned about random variables related to external modules.

**Theorem 1.** Suppose the function $F_{\xi_i} : X(\xi_1, \xi_2) \rightarrow \hat{X}(\xi_1, \xi_2)$ is a linear map from $X(\xi_1, \xi_2)$ to $\hat{X}(\xi_1, \xi_2)$, and is $p$ times differentiable at $\xi_1 = 0$ where $\xi_1$ and $\xi_2$ are independent random variables. Let $X(\xi_1, \xi_2)$ and $X(\xi_1, \xi_2)$ both have PCE representation with order $p$. Then, the evaluation of $F$ can be decomposed into $p + 1$ independent subproblems. A given subproblem $k$ involves the PC terms of $X(\xi_1, \xi_2)$ and the PC terms of $\hat{X}(\xi_1, \xi_2)$ that correspond only to the $k^{th}$ order in $\xi_2$.

**Proof.** Let the PC representation of $X(\xi_1, \xi_2)$ be:

$$X(\xi_1, \xi_2) = \sum_{j=0}^{p} \sum_{i=0}^{p-j} X_{ij} \Psi_i(\xi_1) \Psi_j(\xi_2)$$

(18)

where $\Psi_i(\xi_1)$ and $\Psi_j(\xi_2)$ are the Legendre polynomials of degree $i$ and $j$ for $\xi_1$ and $\xi_2$; and similarly, let the PC representation of $\hat{X}(\xi_1, \xi_2)$ be:

$$\hat{X}(\xi_1, \xi_2) = \sum_{k=0}^{p} \sum_{l=0}^{p-k} \hat{X}_{kl} \Psi_k(\xi_1) \Psi_l(\xi_2)$$

(19)

where $\Psi_k(\xi_1)$ and $\Psi_l(\xi_2)$ are the Legendre polynomials of degree $k$ and $l$ for $\xi_1$ and $\xi_2$.

Then, the coefficients $\hat{X}_{kl}$’s can be computed by:

$$\hat{X}_{kl} = \int_{\Omega} \int_{\Omega} F_{\xi_i}(X(\xi_1, \xi_2)) \Psi_k(\xi_1) \Psi_l(\xi_2) d\xi_1 d\xi_2. \quad (20)$$

Since the operator $F_{\xi_i}$ is linear, it can be decomposed into:

$$F_{\xi_i}(X(\xi_1, \xi_2)) = F_{\xi_i} \left( \sum_{j=0}^{p} \sum_{i=0}^{p-j} X_{ij} \Psi_i(\xi_1) \Psi_j(\xi_2) \right) = \sum_{j=0}^{p} F_{\xi_i} \left( \sum_{i=0}^{p-j} X_{ij} \Psi_i(\xi_1) \Psi_j(\xi_2) \right).$$

(21)
Substituting this into Eq. (20), we obtain:

\[
\hat{X}_{kl} = \sum_{j=0}^{p} \int_{\Omega} \int_{\Omega} F_{\xi_1} \left( \sum_{i=0}^{p-j} X_{ij} \Psi_i (\xi_1) \Psi_j (\xi_2) \right) \Psi_k (\xi_1) \Psi_l (\xi_2) d\xi_1 d\xi_2.
\]  \tag{22}

Since \( F_{\xi_1} \) does not depend on \( \xi_2 \) (i.e., \( F_{\xi_1} \) is random only in \( \xi_1 \)), the terms in the summation corresponding to \( j \neq l \) will vanish giving the following simplified form:

\[
\hat{X}_{kl} = \int_{\Omega} \int_{\Omega} F_{\xi_1} \left( \sum_{i=0}^{p-l} X_{il} \Psi_i (\xi_1) \right) \Psi_k (\xi_1) \Psi_l (\xi_2) d\xi_1 d\xi_2
\]

\[
= \langle \Psi_i^2 \rangle \int_{\Omega} F_{\xi_1} \left( \sum_{i=0}^{p-l} X_{il} \Psi_i (\xi_1) \right) \Psi_k (\xi_1) d\xi_1.
\]  \tag{23}

\[
= \langle \Psi_i^2 \rangle \int_{\Omega} F_{\xi_1} \left( \sum_{i=0}^{p-l} X_{il} \Psi_i (\xi_1) \right) \Psi_k (\xi_1) d\xi_1.
\]  \tag{24}

It can be observed from Eq. (24) that computing \( \hat{X}_{kl} \) only depends on \( X_{il} \) for \( i = 0 \) to \( p - l \).

**Corollary 1.** Suppose the function \( F_{\xi_1} : X(\xi_1, \xi_2) \rightarrow \hat{X}(\xi_1, \xi_2) \) is a linear map from \( X(\xi_1, \xi_2) \) to \( \hat{X}(\xi_1, \xi_2) \), and is \( p \) times differentiable at \( \xi_1 = 0 \) where \( \xi_1 \) and \( \xi_2 \) are independent random variables with respective dimensions \( m_1 \) and \( m_2 \). Let \( X(\xi_1, \xi_2) \) and \( \hat{X}(\xi_1, \xi_2) \) both have polynomial chaos (PC) representations with order \( p \). Then, the evaluation of \( F \) can be decomposed into \( (m_2 + p)!/(m_2!p!) \) independent subproblems. A given subproblem \( k \) involves the PC terms of \( X(\xi_1, \xi_2) \) and the PC terms of \( \hat{X}(\xi_1, \xi_2) \) that correspond only to the \( k \)th term in the single-index PC representation of \( \xi_2 \).

3.3. Applicability to nonlinearly coupled modules

Since non-intrusive sampling is trivially applicable for both linear and nonlinear equations, we will focus on intrusive PCE within our modularly hybrid framework. The methodology for propagating global uncertainty through PCE-based modules with nonlinear partial differential equations, as opposed to those with linear equations, requires generating a sample with the random variables external to the modules and running the sample on the user-provided stochastic nonlinear solver. This amounts to conditional evaluation of the global uncertainty representation with respect to external random variables. In other words, each sampled value of the external variables is used to construct a set of PCE coefficients with respect to the internal random variables. Hence, each subproblem corresponds to an independent evaluation of PCE terms. So if there are \( n_k \) sampled values of the external random variables, then we will end up with \( n_k \) sets of “partial” PCE coefficients, which are then assembled to recover the global uncertainty representation. Here, \( n_k \) needs to be sufficiently large to allow for unique reconstruction of the global representation. In summary, \( R_{ki} \) corresponds to the restriction of PCE format to the internal variables by sampling the external variables, and \( P_{ki} \) corresponds to the global PCE reconstructions from the partial PCE representations. Further details are outside the scope of this paper, as this work focuses on the application of hybrid methods for linear problems.

3.4. Applicability to modules with high dimensions

Our modularly hybrid framework can also be applied to multi-physics stochastic equations involving spatially distributed random parameters (i.e., \( \mathbf{x} \) with spatial dependence). To do so, we
would simply need another specialized module whose solver is equipped to handle uncertainty propagation for spatially distributed random parameters. In principle, spatial dependence increases the dimensionality of the random variable space, so our concepts of hybrid modularization (i.e., sub-problem decomposition, restriction and prolongation) naturally apply to the spatial case as well. In essence, a spatial module can be construed as a high-dimensional linear module. To address the issue of high dimensionality, Karhunen-Loeve expansion (KLE) [25, 26] can be leveraged to decorrelate the spatial process for dimension reduction. KLE has the attractive property that the mean-square error introduced by truncating the expansion is minimized. In addition, one can also decide on the number of terms to truncate based on an acceptable threshold for the total variance of the truncated expansion. For these reasons, KLE has been widely applied to problems involving spatial heterogeneity, and is becoming more prevalent in environmental applications involving subsurface flow problems [27, 28, 29, 30, 31]. For input spatial process with relative low correlation length, the dimensionality of the random input represented by the truncated KLE can be very large. To deal with the “curse of dimensionality”, we apply non-parametric screening method [32] to efficiently identify the most sensitive random variables for truncated KLE with many random variables. Other dimension-reduction techniques have been proposed, such as (1) multi-element, generalized polynomial chaos [33]; and (2) decomposition methods motivated by analysis of variance (ANOVA) techniques [34] and used for high-dimensional UQ problems in [35, 36].

4. Modularly hybrid UQ framework: application

In this section, we demonstrates how our proposed framework can be used in the UQ analyses of sequential multi-step transport reaction modules.

4.1. Two-dimensional transport coupled with first-order reactions

Sequential multi-step reaction models are central to the understanding of various biologic and chemical processes that occur in the subsurface [37, 38]. For example, during denitrification (process used to remove nitrogen from sewage and wastewater), nitrate reacts to produce nitrite, and subsequently ammonia or nitrogen gas [39]. Similarly, reductive anaerobic degradation of tetrachloroethylene (common soil contaminant) to trichloroethylene, to dichloroethylene, and eventually to vinyl chloride, may be modeled using a sequential first-order degradation kinetic model [40]. Being able to apply UQ on these types of sequential multi-step reaction models has tremendous potential to facilitate assessment and control of various biochemical processes that greatly impact the environment.

In the latter case, reductive anaerobic degradation falls under the general category of multi-species reactive transport in porous media, which can be described by the following system of time-dependent partial differential equations:

\[
R_i \frac{\partial c_i}{\partial t} = D_x \frac{\partial^2 c_i}{\partial x^2} + D_y \frac{\partial^2 c_i}{\partial y^2} - v_x \frac{\partial c_i}{\partial x} - v_y \frac{\partial c_i}{\partial y} + R_{i-1} k_{i-1} c_{i-1} - R_i k_i c_i, \quad \forall i = 1, 2, \ldots, n
\]  

(25)

where \( c_i \) [ML^{-3}] is the concentration of the \( i \)th species; \( t \) is time [T]; \( v_x \) and \( v_y \) [LT^{-1}] are the velocity components for \( x \)- and \( y \)-direction respectively; \( D_x \) and \( D_y \) [L^2T^{-1}] are the dispersion coefficients for \( x \)- and \( y \)-direction respectively; \( k_i \) is the first-order reaction rate [T^{-1}]; \( R_i \) is the retardation factor of the \( i \)th species with \( R_0 = 0 \); and \( n \) is the total number of species. Dividing both sides by \( R_i \), we obtain:
\[
\frac{\partial c_i}{\partial t} = \frac{D^x}{R_i} \frac{\partial^2 c_i}{\partial x^2} + \frac{D^y}{R_i} \frac{\partial^2 c_i}{\partial y^2} - \frac{v^x}{R_i} \frac{\partial c_i}{\partial x} - \frac{v^y}{R_i} \frac{\partial c_i}{\partial y} + R_i \left(k_i c_i - R_{i-1} k_{i-1} c_{i-1} - k_i c_i\right), \quad \forall i = 1, 2, \ldots, n. \quad (26)
\]

In matrix form, the system of multi-species equations can be rewritten as:

\[
\frac{\partial c}{\partial t} = D^x \frac{\partial^2 c}{\partial x^2} + D^y \frac{\partial^2 c}{\partial y^2} - V^x \frac{\partial c}{\partial x} - V^y \frac{\partial c}{\partial y} + Ac \quad (27)
\]

where:

\[
c = \begin{pmatrix} c_1 & \ldots & c_n \end{pmatrix}^T,
\]

\[
\begin{pmatrix} D^x \\ D^y \\ V^x \\ V^y \end{pmatrix} = \begin{pmatrix} D^x / R_1 & \ldots & D^x / R_n \\ D^y / R_1 & \ldots & D^y / R_n \\ v^x / R_1 & \ldots & v^x / R_n \\ v^y / R_1 & \ldots & v^y / R_n \end{pmatrix}, \quad (28)
\]

and the structure of the matrix \(A\) depends on the reaction network [41].

In this work, we apply Eq. (27) to model the sequential first-order network that represents the biodegradation of trichloroethylene (TCE). TCE reacts to produce daughter species, dichloroethylene (DCE), while DCE further reacts to produce vinyl chloride (VC), and finally VC reacts to produce ethylene (ETH). That is, there are four species (i.e., \(n = 4\)) and the process is depicted as follows:

\[
\text{TCE} \xrightarrow{l_1}{k_1} \text{DCE} \xrightarrow{l_2}{k_2} \text{VC} \xrightarrow{l_3}{k_3} \text{ETH} \xrightarrow{l_4}{k_4} \quad (29)
\]

where \(l_i\) is the yield coefficient of the \(i\)th reaction, with \(l_i = 1\) for unimolecular reactions. Following Eq. (27), the reaction matrix \(A\) for this network has the following form:

\[
A = \begin{pmatrix}
-k_1 & 0 & 0 & 0 \\
R_1 l_1 k_1 & -k_2 & 0 & 0 \\
0 & R_2 l_2 k_2 & -k_3 & 0 \\
0 & 0 & R_3 l_3 k_3 & -k_4
\end{pmatrix}, \quad (30)
\]

To solve Eq. (27), one popular solution approach is to use “operator splitting”, namely, to split the solution process in two stages to mimic the equation structure which consists of independent transport and reaction terms. At each time step, the “transport equation” is solved for the species concentration, followed by the solution of the “reaction equation” solved by an ODE solver. In the special case of sequential networks with first-order reaction, the reaction equation has an analytical solution [41].

Under operator splitting of Eq. (27), the (vectorized) transport equation is:

\[
\frac{\partial c^{t+\frac{1}{2}}}{\partial t} = D^x \frac{\partial^2 c^{t+\frac{1}{2}}}{\partial x^2} + D^y \frac{\partial^2 c^{t+\frac{1}{2}}}{\partial y^2} - V^x \frac{\partial c^{t+\frac{1}{2}}}{\partial x} - V^y \frac{\partial c^{t+\frac{1}{2}}}{\partial y} \quad (31)
\]

which consists of four independent equations, one per species, since \(D^x, D^y, V^x\) and \(V^y\) are all diagonal matrices. Then, we apply \(c^{t+1/2}\) (i.e., the solution to Eq. (31)) as the initial conditions
to the (vectorized) reaction equation:

\[
\frac{\partial \mathbf{c}^{t+1}}{\partial t} = \mathbf{A} \mathbf{c}^{t+1}
\]

which can be solved analytically [41].

In reactive transport problems, there are many sources of uncertainties. In our model (cf. Eq. (25)), we treat the following parameters as uncertain: the dispersivities (\(\alpha^x, \alpha^y\)), the velocities (\(v^x, v^y\)), and the first-order reaction rates \(k_i\). The dispersion coefficient can be prescribed as a linear function of dispersivity, i.e., \(D^x = \alpha^x v^x\) and \(D^y = \alpha^y v^y\) in the \(x\)- and \(y\)-direction, respectively. We assume \(\alpha^x, \alpha^y, v^x, v^y\) and \(\{k_i\}_{i=1}^4\) as independent second-order random variables. Hence, the uncertainty of the concentration for the first species involves the uncertain parameters \(\alpha^x, \alpha^y, v^x, v^y\), and \(k_1\). The uncertainty of the concentration for the second species involves the uncertain parameters \(\alpha^x, \alpha^y, v^x, v^y, k_1, k_2\) and \(k_3\). Finally, the uncertainty of the concentration for the third species involves all the uncertain parameters \(\alpha^x, \alpha^y, v^x, v^y, k_1, k_2, k_3\) and \(k_4\). While previous works [42, 43] assumed either the transport parameters \(\alpha^x, \alpha^y, v^x, v^y\) or the reactions parameters \(\{k_i\}_{i=1}^n\) are uncertain, this work is the first in examining the uncertainties in both transport and reaction parameters jointly.

### 4.2. Transport module

The transport system consists of four transport equations can be independently solved for each species. Hence, we will discuss the PCE formulation of a generic species’ scalar transport equation.

4.2.1. Variational formulation

When \(\alpha^x, \alpha^y, v^x\) and \(v^y\) are spatially dependent (i.e., for inhomogeneous domains), there is no exact solution for Eq. (33). Hence, we reformulate the problem in its weak or variational form so we may apply approximation methods to compute its solution.

Let \(\mathcal{V}\) be the set of functionals on \(\Omega\) such that

\[
\mathcal{V} = \{ c \in H^1_0(\Omega) : c = c_d(\mathbf{x}) \text{ on } \Gamma_d \}
\]
where $H^1_0(\Omega)$ with $\Gamma_0 = \emptyset$ is the Sobolev space of square-integrable functionals whose first-order derivatives are also square-integrable. Thus, the problem above can be expressed in the following variational form: “Find $c \in V$ such that

$$ a(c, w) = 0 \quad \forall w \in V $$

(36)

where

$$ a(c, w) = \int_\Omega w(x) \frac{\partial c(x, t)}{\partial t} \, dx + \frac{1}{R} \int_\Omega \left( \alpha_x v_x \frac{\partial w(x)}{\partial x} \frac{\partial c(x, t)}{\partial x} + \alpha_y v_y \frac{\partial w(x)}{\partial y} \frac{\partial c(x, t)}{\partial y} \right) \, dx $$

(37)

and $w(x) \in V$ is a test (or weighting) function.”

4.2.2. Galerkin finite-element approximation

To solve the variational form of the problem, we apply Galerkin finite-element discretization involving $T = \{ \Omega^e_k \}_{k=1}^\eta$, a triangulation of $\Omega$ with $\eta$ non-overlapping triangular elements $\Omega^e_k$. Locally on each element $\Omega^e_k$, we assume the finite element approximation $c_h$ of the functional $c \in V$ is linear.

Let $\mathcal{N}$ is the set of nodes of the finite-element mesh which are not lying on $\Gamma_d$. Denote $V^h$ as the finite-element approximation of $V$ from Eq. (35):

$$ V^h = \text{span} \{ \Phi_i \}_{i \in \mathcal{N}}. $$

(38)

The trial function $c_h$ and the test function $w_h$ are discretized accordingly:

$$ c_h(x, t) = \sum_{i \in \mathcal{N}} c_i(t) \Phi_i(x) \in V^h $$

$$ w_h(x, t) = \sum_{j \in \mathcal{N}} w_j(t) \Phi_j(x) \in V^h $$

(39)

where $\Phi_i(x)$ is the corresponding shape function associated with the nodes, and $c_i(t)$ is the transient nodal values at nodal point $i$.

$c$ and $w$ can now be approximated by their respective finite-element counterparts, $c_h$ and $w_h$. Substituting $c_h$ and $w_h$ into Eqs. (36-37) yields a finite-element approximation of the variational form of the problem:

$$ \sum_{i,j \in \mathcal{N}} (m_{ij} \dot{c}_i + a_{ij} c_i) w_j = 0 $$

(40)

where

$$ m_{ij} = \int_\Omega \Phi_i(x) \Phi_j(x) \, dx $$

(41)
\[ a_{ij} = \frac{1}{R} \int_{\Omega} \left( \alpha^x v^x \frac{\partial \Phi_i(x)}{\partial x} \frac{\partial \Phi_j(x)}{\partial x} + \alpha^y v^y \frac{\partial \Phi_i(x)}{\partial y} \frac{\partial \Phi_j(x)}{\partial y} \right) \, dx \]

\[ - \frac{1}{R} \int_{\Omega} \left( v^x \frac{\partial \Phi_i(x)}{\partial x} \Phi_j(x) + v^y \frac{\partial \Phi_i(x)}{\partial y} \Phi_j(x) \right) \, dx. \] (42)

The single term from Eq. (41) can be rewritten as the element matrix \( M_e \), and the two terms from Eq. (42) can also be rewritten as element matrices \( K^1_e \) and \( K^2_e \) respectively (see Appendix A for details). Subsequently, these element matrices can be assembled into the (full-mesh) global matrices \( M, K^1 \) and \( K^2 \).

Using an appropriate indexing of the nodes in \( N \), the finite-element approximation of the variational form (cf. Eqs. (40-42)) can be rewritten as a set of linear equations for the set of transient nodal values \( c_i \) of \( c^h \), as follows:

\[ \dot{M}c + Kc = 0 \] (43)

where \( K = K^1 - K^2 \), and the dimension of \( c \) equates to the cardinality of \( N \) (i.e., \( \dim(c) = \lvert N \rvert \)). Here, \( \dim(c) \) represents the number of unknowns or degrees of freedom in the finite-element problem.

To solve Eq. (43), we apply the backward Euler method for time integration:

\[ (M + K\Delta t) c^{t+1/2} = MC^t. \] (44)

The global matrices \( M \) and \( K \), individually of size \( \lvert N \rvert \times \lvert N \rvert \) (because they were assembled from the element matrices), are very sparse and can be stored compactly. For example, since we used linear triangular elements in the assembly process, we were able to store these global matrices as \( \lvert N \rvert \times 7 \) matrices.

4.2.3. PCE formulation of the 2-D transport system

So far, we have described how to go about solving the deterministic problem (cf. Eq. (33)), in which \( \alpha^x, \alpha^y, v^x \) and \( v^y \) are assumed fixed. However, in practical applications, \( \alpha^x, \alpha^y, v^x \) and \( v^y \) have uncertainties. Thus, we proceed to describe how to solve the stochastic version of this same problem, where \( \alpha^x, \alpha^y, v^x \) and \( v^y \) are assumed to be independent second-order random variables.

By definition, \( \alpha^x, \alpha^y, v^x \) and \( v^y \) are functions of random event \( \theta \) in an abstract probability space \( (\Theta, \Sigma, P) \):

\[ \alpha^x = \alpha^x(\theta); \quad \alpha^y = \alpha^y(\theta); \quad v^x = v^x(\theta); \quad v^y = v^y(\theta). \] (45)

As a result, the solution (i.e., the concentration field \( c \)) is also random and satisfies almost surely the stochastic problem as follows:

\[ \frac{\partial c(x, \theta, t)}{\partial t} = \frac{\alpha^x(\theta) v^x}{R} \frac{\partial^2 c(x, \theta, t)}{\partial x^2} + \frac{\alpha^y(\theta) v^y}{R} \frac{\partial^2 c(x, \theta, t)}{\partial y^2} \]

\[ - \frac{v^x(\theta)}{R} \frac{\partial c(x, \theta, t)}{\partial x} - \frac{v^y(\theta)}{R} \frac{\partial c(x, \theta, t)}{\partial y} \quad x \in \Omega \] (46)
such that \( c(x, \theta, t) = c_d(x), \ x \in \Gamma_d = \partial \Omega \). The random solution \( c(x, \theta, t) \) lies in the space \( \mathcal{V} \otimes L^2(\Theta, P) \), where \( \otimes \) denotes the Cartesian product, and:

\[
c(\cdot, \theta, t) \in \mathcal{V}, \quad c(x, \cdot, t) \in L^2(\Theta, P)
\]

(47)

Here, the deterministic space for the random solution is \( \mathcal{V} \) (from Eq. (35)) and the stochastic space is \( L^2(\Theta, P) \), the set of second-order random variables.

The stochastic variational formulation of the problem is to find \( c \in \mathcal{V} \otimes L^2(\Theta, P) \) such that:

\[
A(c, w) = E[a(c, w)] = \int_\Theta a(c, w) \, dP(\theta), \quad \forall w(x, \theta) \in \mathcal{V} \otimes L^2(\Theta, P)
\]

by taking the expectation of \( a \) (cf. Eq. (37)), where \( w(x, \theta) \in \mathcal{V} \otimes L^2(\Theta, P) \) is a test (or weighting) function of random event \( \theta \).

Recall the deterministic finite-element space \( \mathcal{V}^h \) defined in Eq. (38). Similarly, the semi-discrete stochastic solution lies in:

\[
c_h(x, \theta, t) = \sum_{i \in \mathcal{N}} c_i(\theta, t) \Phi_i(x) \in (\mathcal{V}^h \otimes L^2(\Theta, P))
\]

(49)

(The solution is semi-discrete because we have not yet discretized the stochastic space \( L^2(\Theta, P) \).)

Next, we apply finite-element approximation to the stochastic variational form of the problem, derived from taking the expectation of the finite-element approximation of the deterministic variational form given in Eq. (40). This involves solving for a set of \( N \) transient random variables \( c_i(\theta, t) \) that satisfy Eq. (49) for \( \forall w_i(\theta) \in L^2(\Theta, P), \ i \in \mathcal{N} \). This is equivalent to solving for \( c_i(\theta, t) \) that satisfy:

\[
\sum_{i, j \in \mathcal{N}} E[(m_{ij} \dot{c}_i(\theta, t) + A_{ij}(\theta) c_i(\theta, t)) w_j(\theta)] = 0
\]

(50)

where

\[
A_{ij}(\theta) \triangleq a_{ij}(\theta)
\]

(51)

based on Eqs. (45) and (47).

To apply PCE-based Galerkin projection (to the finite-element approximation of the stochastic variational form), we introduce the “polynomial chaos space” \( \mathcal{W}_\kappa \) onto which we will be projecting the stochastic space \( L^2(\Theta, P) \):

\[
\mathcal{W}_\kappa \equiv \text{span} \{ \Psi_0, \ldots, \Psi_\kappa \} \subset L^2(\Theta, P)
\]

(52)

where \( \{ \Psi_k \}_{k=0}^\kappa \) are polynomial chaoses of some prespecified order with its type and domain depending on the distributions imposed by the independent second-order random variables \( \alpha^x, \alpha^y, v^x \) and \( v^y \). The number of polynomials in this multi-dimensional expansion is denoted by \( (\kappa + 1) \), which is defined in Eq. (6). Thus, \( \alpha^x, \alpha^y, v^x \) and \( v^y \) can now be approximated by the following truncated \( (\kappa + 1) \)-term PCE:

\[
\alpha^x(\xi, \theta) = \sum_{i=0}^\kappa \alpha^{x_i} \Psi_i(\xi(\theta)) \quad \alpha^y(\xi, \theta) = \sum_{i=0}^\kappa \alpha^{y_i} \Psi_i(\xi(\theta))
\]

\[
v^x(\xi, \theta) = \sum_{i=0}^\kappa v^{x_i} \Psi_i(\xi(\theta)) \quad v^y(\xi, \theta) = \sum_{i=0}^\kappa v^{y_i} \Psi_i(\xi(\theta))
\]

(53)
Analogously, the solution \( c^h(x, \xi, t) \) can be approximated by substituting the corresponding truncated PCE to Eq. (49):

\[
c^h(x, \xi, t) = \sum_{i \in \mathbb{N}} \left( \sum_{k=0}^{\kappa} c_{i, k}(t) \Psi_k(\xi) \right) \Phi_i(x) \in \left( \mathcal{V}^h \otimes \mathcal{W}^\kappa \right). \tag{54}
\]

Similarly, the same can be done for the test functions \( w^h(x, \xi, t) \):

\[
w^h(x, \xi, t) = \sum_{i \in \mathbb{N}} \left( \sum_{k=0}^{\kappa} w_{i, k}(t) \Psi_k(\xi) \right) \Phi_i(x) \in \left( \mathcal{V}^h \otimes \mathcal{W}^\kappa \right). \tag{55}
\]

At this point, we can express the semi-discrete stochastic variational formulation in terms of the expansions for \( \alpha^x, \alpha^y, v^x, v^y, c^h \) and \( w^h \).

Next, we apply stochastic Galerkin projection, which involves solving for the coefficients \( \{c_{i, k}\}_{i \in \mathbb{N}, 0 \leq k \leq \kappa} \) and \( \{w_{j, k}\}_{j \in \mathbb{N}, 0 \leq k \leq \kappa} \) that satisfy:

\[
0 = \sum_{i \in \mathbb{N}} \sum_{k=0}^{\kappa} \left[ \int_{\Omega} \Phi_i(x) \Phi_j(x) \, dx - \frac{1}{R} \sum_{i,j \in \mathbb{N}} \sum_{k=0}^{\kappa} \left( \int_{\Omega} \frac{\partial \Phi_i(x)}{\partial x} \frac{\partial \Phi_j(x)}{\partial x} \, dx \right) c_{i, k} w_{j, k} - \int_{\Omega} \frac{\partial \Phi_i(x)}{\partial y} \frac{\partial \Phi_j(x)}{\partial y} \, dx \right] c_{i, k} w_{j, k} \tag{56}
\]

which was derived by expanding Eq. (50) by applying Eqs. (41), (51), (53), and (54).

Due to the orthogonality of the stochastic expansion bases \( \{\Psi\} \), Eq. (57) can be simplified and rewritten in terms of the stochastic global matrices \( [\mathbf{M}] \) and \( [\mathbf{K}] \):

\[
[\mathbf{M}] \dot{[\mathbf{c}]} + [\mathbf{K}][\mathbf{c}] = 0 \tag{58}
\]

In essence, Eq. (58) is the stochastic version of Eq. (43). Due to the stochastic Galerkin projection, we now have a specific instance of Eq. (43) per each of the \((\kappa + 1)\) stochastic modes that resulted from projection onto the stochastic expansion bases.

Thus, the stochastic global matrices \( [\mathbf{M}] \) and \( [\mathbf{K}] \) are constructed from the deterministic global matrices \( \mathbf{M} \) and \( \mathbf{K} \) (from Eq. (50)) as follows:

\[
[\mathbf{M}] = \begin{pmatrix} \mathbf{M} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{M} \end{pmatrix}, \quad [\mathbf{K}] = \begin{pmatrix} \mathbf{K}_{0, 0} & \cdots & \mathbf{K}_{0, \kappa} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{\kappa, 0} & \cdots & \mathbf{K}_{\kappa, \kappa} \end{pmatrix}. \tag{59}
\]
Moreover, the solution to Eq. (58) (i.e., the stochastic concentration field) is:

\[
[c] = \begin{pmatrix} c_0 & \ldots & c_{\kappa} \end{pmatrix}^T
\]  

(60)

where \( c_k = \begin{pmatrix} c_{1,k} & \ldots & c_{|\mathcal{N}|,k} \end{pmatrix}^T \), \( 0 \leq k \leq \kappa \), denotes the vector of nodal values of the \( k \)th stochastic mode of the solution.

To explain in more details the construction of \( M \) and \( K_{kl} \), we show Eq. (58) in its expanded form:

\[
\begin{pmatrix}
M & \ldots & 0 \\
\vdots & \ddots & \vdots \\
0 & \ldots & M
\end{pmatrix}
\begin{pmatrix}
\dot{c}_0 \\
\vdots \\
\dot{c}_\kappa
\end{pmatrix}
+ \begin{pmatrix}
K_{0,0} & \ldots & K_{0,\kappa} \\
\vdots & \ddots & \vdots \\
K_{\kappa,0} & \ldots & K_{\kappa,\kappa}
\end{pmatrix}
\begin{pmatrix}
c_0 \\
\vdots \\
c_\kappa
\end{pmatrix}
= 0
\]  

(61)

Let \( K_{kl} = K_{kl}^1 - K_{kl}^2 \) for \( 0 \leq k, l \leq \kappa \). The construction of \( M, K_{kl}^1 \), and \( K_{kl}^2 \) is analogous to that in Section 4.2.2; the only difference is that these matrices are now assembled from stochastic element matrices instead of their deterministic counterparts.

Using the backward Euler method, we can discretize Eq. (61) in time and rewrite it as:

\[
\begin{pmatrix}
M + K_{0,0} \Delta t & \ldots & K_{0,\kappa} \Delta t \\
\vdots & \ddots & \vdots \\
K_{\kappa,0} \Delta t & \ldots & M + K_{\kappa,\kappa} \Delta t
\end{pmatrix}
\begin{pmatrix}
\dot{c}_0 \\
\vdots \\
\dot{c}_\kappa
\end{pmatrix}
^t + 1/2
= \begin{pmatrix}
M & \ldots & 0 \\
\vdots & \ddots & \vdots \\
0 & \ldots & M
\end{pmatrix}
\begin{pmatrix}
c_0 \\
\vdots \\
c_\kappa
\end{pmatrix}
^t
\]  

(62)

Assuming uniform probability distributions for the random variables \( \{\alpha^x, \alpha^y, v^x, v^y\} \), we can exploit the orthogonality of Legendre polynomial chaos to approximate the mean and variance of the concentration at each grid point \( i \) by:

\[
\bar{c}_i \left( t + \frac{1}{2} \right) = c_{i,0}^{t + \frac{1}{2}}
\]

(63)

\[
\sigma_i^2 \left( t + \frac{1}{2} \right) = \sum_{k=1}^{\kappa} \left( c_{ik}^{t + \frac{1}{2}} \right)^2
\]

(64)

In summary, after applying stochastic Galerkin projection to our stochastic problem, we arrive at Eq. (62). Eq. (62) represents a set of \( (\kappa + 1) \) coupled deterministic systems of partial differential equations (PDEs) to be solved by preconditioned Krylov methods Golub and Loan [44]. The number of PDE systems, \( (\kappa + 1) = (p + m)! / (p!m!) \), is determined by the number of random variables \( m \) and the desired order \( p \) of the polynomials. Each such PDE system is defined over a grid consisting of \( |\mathcal{N}| \) points. Thus, the total number of algebraic equations encapsulated in Eq. (62) is \( (p + m)! / (p!m!) \times |\mathcal{N}| \).

4.3 Reaction module

While the transport system (cf. Eq. (31)) can be solved independently per species, the reaction system (cf. Eq. (32)) is coupled so it must be solved in its vectorized form. Reproduced here for completeness, the reaction problem is to apply \( c^t + \frac{1}{2} \) (the transport solution) as the initial condition for solving \( c^{t+1} \) in:

\[
\frac{\partial c^{t+1}}{\partial t} = Ac^{t+1}
\]

(65)
where $\mathbf{A}$ depends on the structure of the reaction network, which for our specific TCE biodegradation problem, takes on the form of Eq. (30). The random variables under consideration are the first-order reaction rates $\{k_i\}_{i=1}^{4}$ that comprise $\mathbf{A}$.

The reaction module is implemented using non-intrusive UQ methods. Assuming each $k_i$ is uniformly distributed on a prespecified range, we apply Latin hypercube or sparse grid sampling to derive an ensemble of sample points, then compute $\mathbf{c}^{t+1}$ for each sample point analytically.

Following [41], the analytic procedure starts with a decomposition of $\mathbf{A}$:

$$
\mathbf{A} = \mathbf{S} \Lambda \mathbf{S}^{-1}
$$

where $\Lambda$ is a diagonal matrix with diagonal \((-k_1, -k_2, -k_3, -k_4)\),

$$
\mathbf{S} = \begin{pmatrix}
R_1 & R_2 & R_3 & R_4 \\
\frac{R_1}{R_4} \prod_{i=1}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_2}{R_4} \prod_{i=2}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_3}{R_4} \prod_{i=3}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & 1 \\
\frac{R_1}{R_4} \prod_{i=1}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_2}{R_4} \prod_{i=2}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_3}{R_4} \prod_{i=3}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & 1 \\
\frac{R_1}{R_4} \prod_{i=1}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_2}{R_4} \prod_{i=2}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_3}{R_4} \prod_{i=3}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & 1
\end{pmatrix}
$$

and

$$
\mathbf{S}^{-1} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
\frac{R_1}{R_4} \prod_{i=1}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & 1 & 0 & 0 \\
\frac{R_1}{R_4} \prod_{i=1}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_2}{R_4} \prod_{i=2}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & 1 & 0 \\
\frac{R_1}{R_4} \prod_{i=1}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_2}{R_4} \prod_{i=2}^{3} \frac{l_{i-1}k_{i}}{k_i-1} & \frac{R_3}{R_4} \prod_{i=3}^{4} \frac{l_{i-1}k_{i}}{k_i-1} & 1
\end{pmatrix}
$$

Then, $\mathbf{c}^{t+1}$ is computed as follows:

(i) Compute $\tilde{\mathbf{c}} = \mathbf{S}^{-1} \mathbf{c}^{t+\frac{1}{2}}$, then let $\tilde{\mathbf{c}} = (\tilde{c}_1 \ldots \tilde{c}_4)^T$.

(ii) For $i = 1$ to 4, compute $c_i^* = \tilde{c}_i \exp(-k_i \Delta t)$, then let $\mathbf{c}^* = (c_1^* \ldots c_4^*)^T$.

(iii) Finally, compute $\mathbf{c}^{t+1} = \mathbf{S} \mathbf{c}^*$.

The analytic solver can be implemented as a black box, since the only requirement is that it be used to evaluate $\mathbf{c}^{t+1}$ for each input sample of $\{k_i\}_{i=1}^{4}$. Once all sample points have been evaluated, the sample mean and variance for $\mathbf{c}^{t+1}$ can be computed from Eqs. (11-12).

4.4 Propagation of uncertainties through the transport and reaction modules

To propagate uncertainty information between modules, one needs to decide on a common format with which to represent uncertainties as they are passed from one module to another. In our implementation, we have chosen to use PCE coefficients, but samples may also be a viable alternative.

The full reaction-transport system with embedded UQ can now be solved by calling each module sequentially at each time step. At the end of the simulation, the overall uncertainty and sensitivity information can be extracted from the PC coefficients as prescribed in Section 2. The PC formulation for the transport equation needs to be described in the larger context of the full reaction-transport system. The reason for this requirement is due to the fact that, at the beginning of each time step, the incoming species concentrations encompass uncertainties from both the transport and reaction systems from previous time steps. As such, the transport module needs the knowledge of all uncertain parameters in the entire system, not just the ones solely pertaining to...
transport. But nonetheless, the task of managing the global propagation of uncertainties through transport module can be still handled elegantly so that the transport solver can be developed independently from the reaction module (as validated by the decomposition property given in Section 3.2). The decomposition and recombination of the global PCE information inside the stochastic transport module are achieved through the restriction and prolongation operators available through a generic library of utility functions provided as part of our modularly hybrid UQ framework.

At each time step, we iterate between the transport solve and the reaction solve. Since the transport module is implemented using PCE-based method, its outputs are already expressed in the common PCE format, as \((κ + 1)\) coefficients for each grid point. To propagate these coefficients through the reaction module, an ensemble of at least \((κ + 1)\) sample points is required as input into the reaction module. The sample generation is performed by the framework interface so that the reaction solver does not need the knowledge of all parameters in the system. The sample points are then evaluated, analyzed and converted back to the global PC coefficients by polynomial regression.

Here, we quickly summarize the implementation details for the modularly hybrid framework as applied to the reaction-transport problem. We assume uniform distributions for all random variables under consideration (i.e., \(α^x, α^y, v^x, v^y\) and \(\{k_i\}_{i=1}^4\)), as well as linear triangular elements in our approximations.

### 4.4.1. PCE Implementation for the transport module

The objective is to solve Eq. (62). To do so, we need to form both the “left hand side” and “right hand side” matrices in that equation before applying the equation to a linear equation solver. This equation needs to be instantiated and solved for each species.

1. Enumerate the permutations corresponding to the mapping of the multi-dimensional form of the PC expansion to a single-index form. The resulting number of permutations is equivalent to \((κ + 1)\), which is the number of required polynomials for the stochastic Galerkin projection.

2. Compute the PCE representations of \(α^x, α^y, v^x\) and \(v^y\) in terms of random variables \(ξ\).

3. For each \((k, l)\) of the \((κ + 1) \times (κ + 1)\) blocks:

   (a) Compute the effective dispersion coefficients \(D^x_{kl}\) and \(D^y_{kl}\):

   \[
   D^x_{kl} = \sum_{i=0}^{κ} \sum_{j=0}^{κ} α^x_i v^x_j e_{ijkl} \quad D^y_{kl} = \sum_{i=0}^{κ} \sum_{j=0}^{κ} α^y_i v^y_j e_{ijkl}.
   \]  

   (69)

   (b) Compute the effective velocities \(V^x_{kl}\) and \(V^y_{kl}\):

   \[
   V^x_{kl} = \sum_{i=0}^{κ} v^x_i e_{ikl} \quad V^y_{kl} = \sum_{i=0}^{κ} v^y_i e_{ikl}
   \]  

   where \(e_{s kl} = ⟨ΨsΨkΨl⟩, 0 ≤ s, k, l ≤ κ\), and \(e_{s mkl} = ⟨ΨsΨmΨkΨl⟩, 0 ≤ s, m, k, l ≤ κ\).

4. Form the stochastic element matrices \(M^x\), \(K^x_{kl}\), and \(K^y_{kl}\) by applying Eqs. (A.1), (A.8) and (A.9) respectively.

5. Form \(M\), \(K^x_{kl}\), and \(K^y_{kl}\) by block assembly of \(M^x\), \(K^x_{kl}\), and \(K^y_{kl}\) respectively. Compute \(K_{kl} = K^x_{kl} - K^y_{kl}\) for \(0 ≤ k, l ≤ κ\).
6. Let \([\mathbf{A}]\) denote the “left hand side” matrix of Eq. (62). Form \([\mathbf{A}]\) by block assembly of \([\mathbf{M}]\) and \([\mathbf{K}_{kl}]\) as follows:

\[
[\mathbf{A}]_{kl} = \begin{cases} 
\mathbf{M} + \mathbf{K}_{kl} \Delta t & \text{if } k = l \\
\mathbf{K}_{kl} \Delta t & \text{otherwise}
\end{cases} \tag{71}
\]

7. Denote by \(\mathcal{D}\) the set of nodes lying on \(\Gamma_d\), the portion of the domain boundary with Dirichlet conditions. For \(m \in \mathcal{D}\), specify boundary conditions for each \([\mathbf{A}]\) sub-block, denoted by \(\mathbf{B} = [\mathbf{A}]_{kl} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{N}|}\):

\[
\begin{cases} 
\mathbf{B}_{mm} = 1 & \text{if } k = l \\
\mathbf{B}_{mn} = 0 & \text{if } k = l, n \neq m \\
\mathbf{B}_{mn} = 0 & \text{otherwise}
\end{cases} \tag{72}
\]

8. Let \([\mathbf{b}]\) denote the “right hand side” matrix of Eq. (62). Form \([\mathbf{b}] = (\mathbf{b}_0 \ldots \mathbf{b}_\kappa)^T\) where \(\mathbf{b}_k = \mathbf{M}\mathbf{c}^t\) for \(0 \leq k \leq \kappa\), and \(\mathbf{c}^t\) contains the computed concentrations from the previous time step.

9. For \(l \in \mathcal{D}\), we specify boundary conditions for \([\mathbf{b}]\):

\[
\begin{cases} 
b_{l,k} = (c_d)_l & \text{if } k = 0 \\
b_{l,k} = 0 & \text{otherwise}
\end{cases}
\]

where \((c_d)_l\) is the \(l^{th}\) nodal value of the given function \(c_d(x)\) on \(\Gamma_d\).

10. Finally, we compute \(\mathbf{c}^{t+1/2}\) by solving the spectral linear system:

\[
[\mathbf{A}] \mathbf{c}^{t+1/2} = [\mathbf{b}]
\]

where \(\mathbf{c}^{t+1/2} = (c_{1,k}^{t+1/2} \ldots c_{|\mathcal{N}|,k}^{t+1/2})^T\) for \(0 \leq k \leq \kappa\).

4.4.2. Sampling implementation for the reaction module

The objective is to solve Eq. (65) using \(\mathbf{c}^{t+1/2}\) as the initial condition. We proceed as follows:

1. Generate an ensemble of \((\kappa + 1)\) or more samples of \(\{k_i\}_{i=1}^{14}\).

2. Convert \(\mathbf{c}^{t+1/2}\) from the PCE format to the “ensemble” format. Recall that:

\[
c_{j}^{t+1/2}(\xi) = \sum_{i=0}^{\kappa} c_{j}^{t+1/2}(\xi_i)
\]

for each grid point \(j\). Hence, the PC coefficients \(c_{j}^{t+1/2}\) for each grid point are used with the sample coordinates \(\xi\) (generated by Latin hypercube and optionally sparse sampling in our implementation) to compute \(c_{j}^{t+1/2}(\xi)\).

3. Evaluate each sample point using the analytical formulae given in Section 4.3 with \(c_{j}^{t+1/2}(\xi)\) as the initial condition.

4. Convert the sample outputs \(c_{j}^{t+1/2}(\xi)\) back to PC coefficients using Legendre regression.
5. Numerical study

To validate our modularly hybrid UQ approach, we applied the framework to a reaction-transport system with sequential first-order reactions involving four species. We used a two-dimensional domain of size 50 meters by 30 meters, with 50 evenly spaced elements in $x$-direction and 30 evenly spaced elements in $y$-direction. Initial concentrations for all four species were assumed to be zero throughout the computational domain. A Gaussian boundary concentration $c(0, y, t) = \exp\left(\frac{-(y - 15)^2}{45}\right)$ was imposed at the inlet of $x$-direction. The retardation coefficients and yield coefficients in the reaction matrix (cf. Eq. (30)) are treated as deterministic and are set to $R = (2.9, \ 2.8, \ 1.4, \ 5.3)$, and $I = (0.7927 \ 0.7385 \ 0.6458 \ 0.4516)$. In this test scenario, we have $n_x = 51$ grid points in the $x$-direction and $n_y = 31$ grid points in the $y$-direction; the time step is set to $\Delta t = 1$ (day). We used the HYPRE iterative solver package [45] for solving the matrix equations required to compute the deterministic and stochastic finite-element solutions.

We assumed that the velocity in $x$-direction was dominant over the $y$-direction, so the randomness in the $y$-direction was negligible and $v^y = 0$. Subsequently, we have $\{\alpha^x, \alpha^y, v^x, k_1, k_2, k_3, k_4\}$ as our random variables, which we imposed as uniformly distributed with ranges given in Table 1. Each of these variables are mapped onto the Legendre interval $[-1, 1]$. We denote this transformed variable as $\xi$. For example, $v^x \sim U([0.4, 0.6])$, so setting $\gamma^x_v = 0.5$ and $\beta^x_v = 0.1$ would ensure that $\xi$ is mapped back to $v^x$ via $v^x(\xi) = \gamma^x_v + \beta^x_v \xi$. Since there are actually six independent variables ($\alpha^y$ is $\alpha^x$ scaled), there are also six transformed variables $\xi = \{\xi_1, \xi_2, \xi_3, \xi_4, \xi_5, \xi_6\}$.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Distribution</th>
<th>Scale</th>
<th>Randomness</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v^x$ (m d$^{-1}$)</td>
<td>0.4</td>
<td>0.6</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_1$</td>
</tr>
<tr>
<td>$\alpha^x$ (m)</td>
<td>8.0</td>
<td>12.0</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_2$</td>
</tr>
<tr>
<td>$\alpha^y = 0.1 \alpha^x$ (m)</td>
<td>0.8</td>
<td>1.2</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_2$</td>
</tr>
<tr>
<td>$k_1$ (day$^{-1}$)</td>
<td>0.04</td>
<td>0.06</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_3$</td>
</tr>
<tr>
<td>$k_2$ (day$^{-1}$)</td>
<td>0.024</td>
<td>0.036</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_4$</td>
</tr>
<tr>
<td>$k_3$ (day$^{-1}$)</td>
<td>0.016</td>
<td>0.024</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_5$</td>
</tr>
<tr>
<td>$k_4$ (day$^{-1}$)</td>
<td>0.004</td>
<td>0.006</td>
<td>Uniform</td>
<td>Linear</td>
<td>$\xi_6$</td>
</tr>
</tbody>
</table>

As basis for comparison, we approximated the ground truth by quasi-Monte Carlo sampling (sample size of 1000) using a non-intrusive UQ software called PSUADE [46]. We studied the convergence of our method with respect to the polynomial order by examining the concentration mean and standard deviation at $c(25, 15)$, which corresponds to the output at the center of the physical domain. We observed from Fig. 1 that the polynomial order $p = 3$ provides reasonable accuracy for this problem.

For subsequent experiments, we used the “optimal” polynomial order of $p = 3$ for the PCE implementation. Figs. 2 and 3 show the computed means $\mu$ and standard deviations $\sigma$ for the species concentrations at $t = 30$ days. Fig. 2 shows, for each species, the uncertainty bands of $\mu \pm \sigma$ along the center line $y = 15$. Fig. 3 shows $\mu$ and $\sigma$ over the entire two-dimensional domain. Lastly, we compare these results against the ground truth (approximated by sampling) in Fig. 4.

Recall from Section 4.1 that, due to the structure of the sequential network, subsequent species’ concentrations depend on incrementally more variables than the precedent species. In particular, species 1 depends on three random variables ($\{v_x, \alpha_x, k_1\}$); species 2 depends on four ($\{v_x, \alpha_x, k_1, k_2\}$);
Figure 1: Errors of concentration mean and standard deviation computed by the modularly hybrid UQ method compared to pure sampling using 1000 samples. Concentrations for species 1, 2, 3 and 4 are shown at top left, top right, bottom left and bottom right respectively. All concentrations shown are taken from the center of the physical domain at $t = 30$ days.
Figure 2: Uncertainty bands ($\mu \pm \sigma$) derived from the concentration mean $\mu$ and the standard deviation $\sigma$ computed by the modular hybrid UQ method using $p = 3$ for the PCE order. Concentrations for species 1, 2, 3 and 4 are shown at top left, top right, bottom left and bottom right respectively. All concentrations shown are taken from the center line $y = 15$ of the physical domain at $t = 30$ days.

Figure 3: Contours of the concentration mean $\mu$ and the standard deviation $\sigma$ computed by the modular hybrid UQ method using $p = 3$ for the PCE order. Concentrations for species 1, 2, 3 and 4 are shown at top left, top right, bottom left and bottom right respectively. All concentrations are from $t = 30$ days.
Figure 4: Concentration mean and standard deviation computed by the modularly hybrid UQ method (using $p = 3$ for the PCE order) compared to pure sampling using 1000 samples. Concentrations for species 1, 2, 3 and 4 are shown at top left, top right, bottom left and bottom right respectively. All concentrations shown are taken from the center line $y = 15$ of the physical domain at $t = 30$ days.

species 3 depends on five ($\{v_x, \alpha_x, k_1, k_2, k_3\}$); and species 4 depends on six ($\{v_x, \alpha_x, k_1, k_2, k_3, k_4\}$).

If one were to apply third-order PCE to the full problem, in order to capture the uncertainties at each grid point, according to Eq. (6):

- species 1 would require $(3 + 3)!/(3!3!) = 20$ terms;
- species 2 would require $(4 + 3)!/(4!3!) = 35$ terms;
- species 3 would require $(5 + 3)!/(5!3!) = 56$ terms;
- species 4 would require $(6 + 3)!/(6!3!) = 84$ terms.

Although our method does not solve the problem using 20-, 35-, 56- and 84-terms PCE, our method does use this format to store the global uncertainty information from both modules. Recall that inference of $v_x$ and $\alpha_x$ takes place in the transport module (via third-order PCE) and inference of $\{k_i\}_{i=1}^4$ takes place in the reaction module (via sampling). To match the incoming 20-, 35-, 56- and 84-terms PCE to the transport module’s internal format, the incoming uncertainty information from the reaction module are decomposed into subproblems. Let $m_1$ denote the number of variables in the transport module, $m_2$ denote the number of variables in the reaction module, and $m = m_1 + m_2$ denote the number of joint variables. Each species’ $(m + p)!/(m!p!)$-terms PCE will be decomposed into $((m_2 + p)!/m_2!p!)$ subproblems, one for each term in the single-index PC representation of the reaction variables. For example, species 1 has $m_1 = 2$ transport variables and $m_2 = 1$ reaction variable. Its 20-terms PCE will be decomposed into 4 smaller subproblems. The first subproblem corresponds to zeroth order of $k_1$, which contains $(m_1 + p)!/(m_1!p!) = 10$ terms.
involving only \( v_x \) and \( \alpha_x \). The second subproblem corresponds to first order of \( k_1 \), which contains \((m_1 + p - 1)!/m_1!(p - 1)! = 6 \) terms. The third subproblem corresponds to second order of \( k_1 \), which contains \((m_1 + p - 2)!/m_1!(p - 2)! = 3 \) terms. The fourth subproblem corresponds to third order of \( k_1 \), which contains \((m_1 + p - 3)!/m_1!(p - 3)! = 1 \) term. Similarly:

- species 2’s 35-terms PCE will be decomposed into 10 subproblems (one containing 10-terms, two 6-terms, three 3-terms, and four 1-term);
- species 3’s 56-terms PCE into 20 subproblems (one containing 10-terms, three 6-terms, six 3-terms, and ten 1-term);
- species 4’s 84-terms PCE into 35 subproblems (one containing 10-terms, six 6-terms, three 3-terms, and twenty 1-term).

These subproblems are then solved by the transport module, then the results are subsequently re-packaged back into the 20-, 35-, 56- and 84-terms PCE format (for each of four species, respectively) for subsequent processing by the reaction module. For the reaction module, the incoming uncertainty information (in PCE format) would have 84 coefficients for each grid point. Thus, to propagate and reconstruct the coefficients, a minimum sample size of 84 is required as input into the reaction module.

6. Conclusion

Intrusive UQ methods offer mathematically rigorous and also potentially highly efficient means of investigating and quantifying uncertainties in simulation models. However, the complexities of their formulation and implementation have hampered the practicality of these method for large-scale multi-physics applications. In this paper, a flexible modularly hybrid UQ methodology for encapsulating mixed intrusive/non-intrusive UQ approach for multi-physics applications is proposed. The vision is that, by providing a generic computational framework and the associated algorithms to manage the complexities of global uncertainty/sensitivity propagation, this approach can streamline the development and maintenance of UQ methods on a per-module basis, allowing the application developers to swap in/out new UQ methods for any modules without the need to worry about incompatibilities with other modules. This vision is compatible with modern day “plug-and-play” philosophy. We have developed a computational framework for this approach and applied it to propagate uncertainties and global sensitivities for a multi-species problem involving diffusive transport and sequential first-order reactions, where intrusive PCE is available for transport and only a deterministic solver is available for reaction.

Since our flexible UQ uses a global PCE representation, it inherits the shortcomings of PCE-based methods, such as difficulties in handling nonlinearities; discontinuities; long time integration; and curse of dimensionality (namely, that the computational complexity grows exponentially with the total number of stochastic variables). To deal with discontinuities and long time integration, one approach is to use multi-element generalized polynomial chaos [36] into our hybrid UQ framework by multiple hybrid UQ instantiation on the elements in parallel. To deal with high dimensionality arise from mild spatial heterogeneity, an approach is to use truncated KLE with non-intrusive non-parametric screening method [32]. To deal with strong or multi-layer spatial heterogeneity, we will consider extending our current framework to a multi-scale flexible UQ framework using random domain decomposition [24].
Acknowledgements

This research was funded by U. S. Department of Energy Office of Advanced Scientific Computing Research Applied Mathematics Program and performed under the auspices of the U. S. Department of Energy by Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344.

Appendix A. Element matrices

In this appendix, we provide the formula for computing the element matrices $M^e$, $K_1^e$, and $K_2^e$. $M^e$ is derived from Eq. (41) as follows:

$$
M^e = \int_{\Omega^e} \Phi_i(x) \Phi_j(x) \, dx = \int_{\Omega^e} \begin{bmatrix} 
\Phi_1 \\
\Phi_2 \\
\Phi_3 
\end{bmatrix} \begin{bmatrix} 
\Phi_1 & \Phi_2 & \Phi_3 
\end{bmatrix} \, dx = \frac{A}{12} \begin{pmatrix} 
2 & 1 & 1 \\
1 & 2 & 1 \\
1 & 1 & 2 
\end{pmatrix}
$$

(A.1)

where the $\Phi_i(x,y)$ is the shape function for the linear triangular element:

$$
\Phi_1 = \frac{1}{2A} [(x_2 y_3 - x_3 y_2) + (y_2 - y_3) x + (x_3 - x_2) y]
$$

(A.2)

$$
\Phi_2 = \frac{1}{2A} [(x_3 y_1 - x_1 y_3) + (y_3 - y_1) x + (x_1 - x_3) y]
$$

(A.3)

$$
\Phi_3 = \frac{1}{2A} [(x_1 y_2 - x_2 y_1) + (y_1 - y_2) x + (x_2 - x_1) y]
$$

(A.4)

and

$$
A = \begin{vmatrix} 
1 & x_1 & y_1 \\
1 & x_2 & y_2 \\
1 & x_3 & y_3 
\end{vmatrix}
$$

(A.5)

is the area of each triangular element, where $x_i$ and $y_i$ be the coordinate value at the $i^{th}$ node.

$K_1^e$ and $K_2^e$ are derived from the two terms in Eq. (42):

$$
K_1^e = (K_1^e)_{ij} = \frac{1}{R} \int_{\Omega^e} \left( \alpha x^2 \frac{\partial \Phi_i}{\partial x} \frac{\partial \Phi_j}{\partial x} + \alpha y^2 \frac{\partial \Phi_i}{\partial y} \frac{\partial \Phi_j}{\partial y} \right) \, dx \quad 1 \leq i, j \leq 3.
$$

(A.6)

$$
K_2^e = (K_2^e)_{ij} = \frac{1}{R} \int_{\Omega^e} \left( \nu \frac{\partial \Phi_i}{\partial x} + \nu \frac{\partial \Phi_j}{\partial y} \right) \, dx \quad 1 \leq i, j \leq 3.
$$

(A.7)

which simplify to:

$$
K_1^e = \frac{1}{4AR} \begin{pmatrix} 
\alpha x^2 a_1^2 + \alpha y^2 b_1^2 & \alpha x^2 a_1 a_2 + \alpha y^2 b_1 b_2 & \alpha x^2 a_1 a_3 + \alpha y^2 b_1 b_3 \\
\alpha x^2 a_1 a_2 + \alpha y^2 b_1 b_2 & \alpha x^2 a_2^2 + \alpha y^2 b_2^2 & \alpha x^2 a_2 a_3 + \alpha y^2 b_2 b_3 \\
\alpha x^2 a_1 a_3 + \alpha y^2 b_1 b_3 & \alpha x^2 a_2 a_3 + \alpha y^2 b_2 b_3 & \alpha x^2 a_3^2 + \alpha y^2 b_3^2 
\end{pmatrix}
$$

(A.8)

$$
K_2^e = \frac{1}{6R} \begin{pmatrix} 
v^2 a_1 + v^y b_1 & v^2 a_2 + v^y b_2 & v^2 a_3 + v^y b_3 \\
v^2 a_1 + v^y b_1 & v^2 a_2 + v^y b_2 & v^2 a_3 + v^y b_3 \\
v^2 a_1 + v^y b_1 & v^2 a_2 + v^y b_2 & v^2 a_3 + v^y b_3 
\end{pmatrix}
$$

(A.9)

where $a_1 = y_1 - y_3$, $a_2 = x_3 - x_1$, $a_3 = x_1 - x_2$, $b_1 = x_3 - x_2$, $b_2 = x_1 - x_3$ and $b_3 = x_2 - x_1$. 27
References


