An Empirical Chaos Expansion Method for Uncertainty Quantification

Melvin Leok and Gautam Wilkins

Abstract. Uncertainty quantification seeks to provide a quantitative means to understand complex systems that are impacted by parametric uncertainty. The polynomial chaos method is a computational approach to solve stochastic partial differential equations (SPDE) by projecting the solution onto a space of orthogonal polynomials of the stochastic variables and solving for the deterministic coefficients. Polynomial chaos can be more efficient than Monte Carlo methods when the number of stochastic variables is low, and the integration time is not too large. When performing long-term integration, however, achieving accurate solutions often requires the space of polynomial functions to become unacceptably large. This paper presents an alternative approach, where sets of empirical basis functions are constructed by examining the behavior of the solution for fixed values of the random variables. The empirical basis functions are evolved over time, which means that the total number can be kept small, even when performing long-term integration. We introduce this method of empirical chaos expansion, and apply it to a number of model equations, demonstrating that the computational time scales linearly with the final integration time. That is not the case for polynomial chaos in general, since achieving accuracy for long-term integration usually requires larger polynomial bases, causing a nonlinear scaling with the final integration time. We also present an analytical method that uses the dynamics of the SPDE to evolve the empirical basis functions and demonstrate how it can be applied to extend the validity of empirical basis functions in time without the need to sample additional realizations of the original SPDE.

Keywords: Uncertainty Quantification, Stochastic Galerkin Method, Generalized Polynomial Chaos, Empirical Basis Functions, Proper Orthogonal Decomposition.

1. Introduction

Consider the stochastic initial boundary value partial differential equation (SPDE) model problem,

$$u_t(x,t,\xi) = L(u,x,t,\xi), \tag{1.1}$$

where x is a spatial variable, t is time, ξ is a random variable with known distribution on probability space Ω, and L is a linear or nonlinear differential operator. SPDEs are used to model systems that contain small scale stochastic components along with large scale deterministic components. Frequently, the deterministic components arise from the governing physics, while the stochastic components are due to measurement errors or some other form of underlying uncertainty. Assuming that the distribution of the stochastic variables is known, we wish to predict the distribution of the solution u. SPDEs have frequently demonstrated their use in modeling physical phenomena such as wave propagation [**40, 24**], diffusion [**42, 41, 21**], Burgers and Navier–Stokes equations with random forcing [**2, 52, 54, 53, 34, 7, 6, 8, 47, 46**], multivariable predictive control [**36, 30, 31, 32**], and chemical reactors with uncertainties [**49**].

1.1. Polynomial Chaos.

The original polynomial chaos formulation was introduced by Wiener [**55, 56**], who used Hermite polynomials to model a Gaussian random process. Polynomial chaos methods begin by choosing a polynomial basis that is orthonormal with respect to the distribution of the random variable ξ. Such polynomial bases are known for standard distributions. If we let $\{P_i\}_{i=1}^{\infty}$ be the orthogonal polynomial

2000 *Mathematics Subject Classification.* Primary .

basis functions, then we can project the solution u onto the basis functions as follows,

$$\hat{u}^i(x,t) = \langle u, P_i \rangle \coloneqq \int_\Omega u(x,t,\xi) P_i(\xi) d\mu,$$

where μ is the measure of the probability space Ω. We can then represent the solution u by

$$u(x,t,\xi) = \sum_{i=1}^\infty \hat{u}^i(x,t) P_i(\xi). \tag{1.2}$$

The problem of computing $u(x,t,\xi)$ is equivalent to determining the expansion coefficients $\hat{u}^i(x,t)$. Projecting the initial conditions for $u(x,t,\xi)$ onto the basis functions yield the initial conditions for $\hat{u}^i(x,t)$. Substituting the expansion (1.2) into the model problem (1.1) yields

$$\sum_{i=1}^\infty \hat{u}^i_t(x,t) P_i(\xi) = L\left(\sum_{i=1}^\infty \hat{u}^i(x,t) P_i(\xi), x, t, \xi\right).$$

We can then multiply by a test function P_j and take an expectation over Ω to obtain

$$\hat{u}^j_t(x,t) = E\left[\sum_{i=1}^\infty \hat{u}^i_t(x,t) P_i(\xi) P_j(\xi)\right] = E\left[L\left(\sum_{i=1}^\infty \hat{u}^i(x,t) P_i(\xi), x, t, \xi\right) P_j(\xi)\right],$$

where the expectation in the middle simplifies due to the P_i's being orthonormal. We can then truncate the infinite expansion on the right hand side at some finite value N, to get

$$\hat{u}^j_t(x,t) E\left[L\left(\sum_{i=1}^N \hat{u}^i(x,t) P_i(\xi), x, t, \xi\right) P_j(\xi)\right], \quad 1 \le j \le N. \tag{1.3}$$

In practice, the right hand side can usually be simplified as well, but it depends on the form of the differential operator L. This results in a system of coupled deterministic differential equations for the unknown $\hat{u}^i$'s. Solving the system will give an approximation of the true solution u, and this is known as the *stochastic Galerkin method* [**17**].

A result by Cameron and Martin [**3**] established that the series (1.2) converges strongly to L_2 functionals, which implies that the series converges for stochastic processes with finite second moments. In [**17**], Hermite polynomial expansions were coupled with finite element methods to solve SPDEs, and the method was applied to model uncertainty in a variety of physical systems [**15, 48, 61, 9**], and the convergence properties of polynomial chaos methods for various SPDEs was studied [**33, 5, 39, 12**] . An extension by Xiu and Karniadakis [**60**] proposed the use of orthogonal polynomials in the Askey scheme [**1**] for a variety of other random variable distributions, including uniform and beta distributions. The work also established the same strong convergence results that held for Hermite polynomial expansions of normal random variables. This method is known as generalized polynomial chaos (gPC), and an overview is provided in [**58**]. gPC methods have been applied to a variety of problems in uncertainty quantification, including fluid dynamics and solid mechanics [**59, 62, 63, 45, 10, 29, 26**].

Since their introduction, two well-known issues have become apparent with the gPC method. First, if the number of random variables is too large, then the resulting stochastic Galerkin system becomes too expensive to solve, and Monte Carlo methods outperform gPC methods [**50, 18**]. This is the curse of dimensionality, and in this case is due to the need to introduce multidimensional polynomials, one dimension for each random variable. Including even modest orders of high-dimensional polynomials can quickly become prohibitively expensive. Methods have been proposed to limit the growth rate, including sparse truncation [**20**] that selectively eliminates high-order polynomials, modifications of stochastic collocation methods[**11**], multi-element gPC expansions [**50, 51, 13**], and model reduction to project a high-dimensional random space onto a lower-dimensional one while still preserving important dynamics [**10, 16, 37, 38**]. While these methods slow the curse of dimensionality, none of them entirely eliminate it, and for systems with a high number of random variables it is more efficient to turn to Monte Carlo methods, which have a convergence rate independent of the dimension of the random space.

The second issue with the gPC method is that in order to accurately approximate a solution with unsteady dynamics over a long time interval, a large number of basis functions must be used, i.e., we must choose a large value for N in (1.3) [**18, 35, 20**]. This is not particularly surprising, but it is problematic since the amount of work does not scale linearly with the order of the polynomial basis functions; this issue becomes particularly troublesome as the number of random variables increases. While multi-element gPC expansions [**50, 51**] have been suggested as an option, they do not entirely solve this issue. Another approach is the time-dependent polynomial chaos method [**14, 19**], which integrates for a short period of time, then treats the solution as a new random variable and attempts to numerically derive an orthogonal polynomial basis for the new random variable. Later, we will introduce a method that addresses this issue by combining model reduction techniques with gPC, and discuss a basis evolution technique that further reduces the computational cost of the approach.

1.2. General Basis. The above results can be extended in a straightforward manner to non-orthonormal bases as well. Consider any set of functions $\{\Psi^i\}_{i=1}^{\infty}$ that form a basis for $L^2(\Omega)$, but are not necessarily orthonormal. Then, there exist coefficients $\hat{u}^i(x,t)$ such that

$$u(x,t,\xi) = \sum\nolimits_{i=1}^{\infty} \hat{u}^i(x,t)\Psi^i(\xi),$$

which implies that

$$\left\langle u, \Psi^i \right\rangle = \left\langle \sum\nolimits_{j=1}^{\infty} \hat{u}^j \Psi^j, \Psi^i \right\rangle = \sum\nolimits_{j=1}^{\infty} \hat{u}^j \left\langle \Psi^j, \Psi^i \right\rangle.$$

If we let

$$A_{ji} = \left\langle \Psi^j, \Psi^i \right\rangle, \quad \hat{u} = \begin{pmatrix} \hat{u}^1 \\ \hat{u}^2 \\ \vdots \end{pmatrix}, \quad \text{and} \quad f = \begin{pmatrix} \left\langle u, \Psi^1 \right\rangle \\ \left\langle u, \Psi^2 \right\rangle \\ \vdots \end{pmatrix},$$

then we can write the above system as

$$A\hat{u} = f, \tag{1.4}$$

where the $\hat{u}$ vector is unknown, and the A and f matrices can be computed by inner products (if u is known). Since the model problem is an initial boundary value problem, we can compute the initial values for the $\hat{u}_i$'s. Just like with standard polynomial chaos, we can derive a stochastic Galerkin method by substituting the expansion into the model problem (1.1), multiplying by a test function Ψ^j, and taking an expectation over Ω. If we truncate the expansion at finite N, we obtain

$$E\left[\sum\nolimits_{i=1}^{N} \hat{u}_t^i(x,t)\Psi^i(\xi)\Psi^j(\xi)\right] = E\left[L\left(\sum\nolimits_{i=1}^{N} \hat{u}^i(x,t)\Psi^i(\xi), x, t, \xi\right)\Psi^j(\xi)\right]. \tag{1.5}$$

Now, if we let

$$\tilde{A}_{ji} = \left\langle \Psi^j, \Psi^i \right\rangle, \quad \hat{u} = \begin{pmatrix} \hat{u}^1 \\ \hat{u}^2 \\ \vdots \\ \hat{u}^n \end{pmatrix}, \quad \text{and} \quad b_j = E\left[L\left(\sum\nolimits_{i=1}^{N} \hat{u}^i(x,t)\Psi^i(\xi), x, t, \xi\right)\Psi^j(\xi)\right],$$

then, we obtain the system

$$\tilde{A}\hat{u}_t = b, \tag{1.6}$$

which is a deterministic implicit differential equation whose solution is the $\hat{u}^i$ coefficients.

1.3. Conversion Between Bases. Assume we have two distinct bases $\{\Psi^i\}_{i=1}^{\infty}$ and $\{\Phi^i\}_{i=1}^{\infty}$ for $L^2(\Omega)$, such that,

$$u(x,t,\xi) = \sum\nolimits_{i=1}^{\infty} \hat{u}_{\Phi}^i(x,t)\Phi^i(\xi), \qquad u(x,t,\xi) = \sum\nolimits_{i=1}^{\infty} \hat{u}_{\Psi}^i(x,t)\Psi^i(\xi).$$

If we know $\hat{u}_{\Psi}^i$, we can compute $\hat{u}_{\Phi}^i$ with a standard change of basis operation,

$$\left\langle u, \Phi^j \right\rangle = \sum\nolimits_{i=1}^{\infty} \left\langle \hat{u}_{\Psi}^i(x,t)\Psi^i, \Phi^j \right\rangle = \sum\nolimits_{i=1}^{\infty} \hat{u}_{\Psi}^i(x,t) \left\langle \Psi^i, \Phi^j \right\rangle.$$

Letting

$$M_{ji} = \left\langle \Psi^i, \Phi^j \right\rangle, \quad \text{and} \quad b_j = \left\langle u, \Phi^j \right\rangle,$$

implies that

$$b = M\hat{u}_{\Psi}. \tag{1.7}$$

The right hand side is a straightforward computation, since $\hat{u}_{\Psi}$ is known, and we can use (1.4) to compute $\hat{u}_{\Phi}$. The same computation applies when the two bases are finite-dimensional instead of infinite-dimensional.

In the finite-dimensional case, the operations above first project u onto the subspace spanned by $\{\Psi^j\}$, followed by a projection onto the subspace spanned by $\{\Phi^i\}$.

2. Empirical Bases

We now seek to resolve the issue of long-term integration for gPC methods, using the following general approach. First, we come up with a finite and small set of basis functions that result in a low projection error over a short time interval (say $0 \leq t \leq \tau_0$). Then, perform a stochastic Galerkin method via (1.5) in order to approximate the solution up to time τ_0. This will give values for the $\hat{u}^i$ coefficients. Now, come up with a different set of basis functions that result in a low projection error over the short time interval $\tau_0 \leq t \leq \tau_1$. We can use the previous solution at τ_0 as the initial condition, and calculate the initial values of the $\hat{u}^i$'s in the new basis by applying the change of basis equation (1.7). We can then perform another stochastic Galerkin method with the new basis to compute the solution up to time τ_1. Continuing this process iteratively allows us to compute the solution out to a long time t with a low error, and at each individual step, we only require a relatively small basis.

The principal issue is determining a subspace spanned by a small number of basis functions for each interval $\tau_i \leq t \leq \tau_{i+1}$, such that the solution u will have a low projection error in the subspace. Once we have such a set of basis functions for each time interval, then we can follow the approach outlined above to solve the SPDE on a relatively small set of basis functions.

A similar attempt is made in [**14**], but it relies on integrating the solution to a small time in the future, and then treating the solution as a new random variable that must be added to the gPC expansion before integrating out to a later time. This entails numerically deriving a new set of orthogonal polynomial basis functions that are inferred from the numerical probability distribution of the solution. In this paper, we present an alternate approach, where we no longer require the basis functions to be orthonormal, and instead use techniques of model reduction to construct an empirically optimal set of basis functions at each given timestep.

2.1. Sampling Trajectories. Fix a single point (x_0, t_0) in spacetime, and examine how the function $u(x_0, t_0, \xi)$ varies in the random variable ξ. If we examine many such sample trajectories, then we can choose a good subspace for projecting the solution by requiring that it have a low projection error for most or all of the sample trajectories.

We can find such trajectories by choosing a set of fixed values for ξ, say $\{\xi_l\}_{l=1}^K$, and then solving the resulting deterministic PDE for each ξ_l, much like in Monte Carlo methods. The difference is that Monte Carlo methods require solving the PDE for an extremely large number of values of ξ before they converge, and we use much fewer sample trajectories. Once we know the solutions for each of the values of $\{\xi_l\}$, we can construct the following matrix,

$$T = \begin{pmatrix} u(x_1,t_1,\xi_1) & u(x_1,t_1,\xi_2) & \dots & u(x_1,t_1,\xi_K) \\ u(x_1,t_2,\xi_1) & u(x_1,t_2,\xi_2) & \dots & u(x_1,t_2,\xi_K) \\ \vdots & \vdots & & \vdots \\ u(x_1,t_N,\xi_1) & u(x_1,t_N,\xi_2) & \dots & u(x_1,t_N,\xi_K) \\ u(x_2,t_1,\xi_1) & u(x_2,t_1,\xi_2) & \dots & u(x_2,t_1,\xi_K) \\ \vdots & \vdots & & \vdots \\ u(x_M,t_N,\xi_1) & u(x_M,t_N,\xi_2) & \dots & u(x_M,t_N,\xi_K) \end{pmatrix}, \tag{2.1}$$

where we have discretized spacetime into the set of points $\{(x_i, t_j)\}_{(i,j)=(1,1)}^{(M,N)}$. Each row of the matrix T represents a discrete trajectory of the solution as a function of the random variable ξ. Note that, in general, this matrix will have far more rows than columns. We now seek a, hopefully small, set of basis functions in ξ that will result in a low projection error for each of the rows of T. The motivating assumption is that such a set of basis functions will also result in a low projection error for the true solution $u(x,t,\xi)$.

2.2. Proper Orthogonal Decomposition. Proper Orthogonal Decomposition (POD) is a standard tool in the field of model reduction [**4, 23**] and is also known as the Karhunen–Loève transform [**22, 25**], or Principal Component Analysis [**44**] in the finite-dimensional case. It is often used to construct low-dimensional representations of very high-dimensional spaces that preserve the most important dynamics [**27, 28, 57**].

Following the exposition from [**27**], in the discrete case we have T in $\mathbb{R}^{NM} \times \mathbb{R}^{K}$ from (2.1), where we view each row of T as a sample from a random vector that takes values in $\mathbb{R}^{K}$. We then seek a set of $k < K$ basis vectors such that the rows of T can be projected with minimum mean square error onto the k basis vectors. The optimization problem can be expressed as

$$\min_{P_k} \; E\left[\|v - P_k v\|^2\right],$$

where v is a random vector from the space sampled by the rows of T, the expectation is over the rows of T, $P_k v$ is the projection of v onto the k basis vectors, and the minimization is over all sets of k basis vectors. In practice, we perform a singular value decomposition (SVD),

$$T = U \Sigma V^T.$$

Then, the first k columns of the right singular matrix V are the optimal set of k basis vectors that minimize the mean square error when we project the NM rows of T onto their span. This will give us a set of k vectors that span a subspace that will result in a low projection error for all of the rows of the T matrix. We can now construct a basis function in ξ for each row of the V matrix by constructing an interpolant through its values at each of the ξ_l points.

2.3. Empirical Chaos Expansion. At this point, we can construct a general algorithm to solve the SPDE model problem (1.1) using successive sets of empirical basis functions. We refer to this method as *empirical chaos expansion*. We begin by choosing a set, $\{\xi_l\}_{l=1}^K$, from the range of the random variable ξ. If there were multiple random variables, we would choose a set of random vectors from the product space of the random variables. If we wish to solve the SPDE from time t_0 to time t_f, we partition the interval by choosing nodal points $t_0 < \tau_1 < \tau_2 < \cdots < \tau_{n-1} < t_f$. We then solve the deterministic partial differential equation for each fixed value of ξ in the set $\{\xi_l\}_{l=1}^K$, starting at time t_0 and ending at time τ_1. Assuming that the solutions are computed using a finite-difference method, we will have values for the solution $u(x, t, \xi)$ at a set of points (x_i, t_j, ξ_l). Fixing a point in spacetime and varying the value of ξ yields a row of the matrix T in Section 2.1, and we use the full set of solutions to construct the full T matrix. We then perform a POD of the T matrix by computing its singular value decomposition, and choosing the first N_1 columns of the right singular value matrix to be the set of basis functions. In practice, N_1 is chosen so that the (N_1+1)-st singular value is small relative to the largest singular value. Each column of the right singular value matrix can be viewed as a table of values for the basis function at each point in $\{\xi_l\}_{l=1}^K$.

We could construct an interpolant for each of the basis column vectors, but it is computationally faster to use a numerical quadrature formula with $\{\xi_l\}_{l=1}^K$ as the quadrature points. We denote the set of empirical basis functions for the time interval $[t_0, \tau_1]$ by $\{\Psi_1^i(\xi)\}_{i=1}^{N_1}$. For the first time interval $[t_0, \tau_1]$, we can approximate the true solution u by $u(x, t, \xi) \approx \sum_{i=1}^{N_1} \hat{u}_1^i(x, t)\Psi_1^i(\xi)$. To determine the initial values of $\hat{u}_1^i$, we project the initial condition onto each of the basis functions, $\hat{u}_1^i(x, t_0) = \left\langle u(x, t_0), \Psi_1^i \right\rangle$, and we solve the propagation equation (1.6) for the stochastic Galerkin system for the SPDE, which computes $\hat{u}_1^i$ up to time τ_1. The expectations in the propagation equation can be computed using numerical quadrature.

The solution at the final time τ_1 can be used as the initial condition on the interval $[\tau_1, \tau_2]$. We begin by computing the solution to the deterministic partial differential equation for each fixed value of ξ in the set $\{\xi_l\}_{l=1}^K$, starting at time τ_1 and ending at time τ_2. We now use the computed solutions to construct a new T matrix that has trajectories over the time interval $[\tau_1, \tau_2]$. We then perform a POD of the T matrix by computing its singular value decomposition, and choose the first N_2 columns of the right singular value matrix to be the set of basis functions. This gives a set of empirical basis functions $\{\Psi_2^i(\xi)\}_{i=1}^{N_2}$. Although we have values $\hat{u}_1^i(x, t)$ up to time τ_1, these are the coefficients in the old basis $\{\Psi_1^i(\xi)\}_{i=1}^{N_1}$. We can convert them to coefficients in the new basis $\{\Psi_2^i(\xi)\}_{i=1}^{N_2}$ by following the method described in Section 1.3. Recall that applying the method requires us to compute entries of $M_{ji} = \left\langle \Psi_1^i, \Psi_2^j \right\rangle$. Again, these inner products can be computed using numerical quadrature. Then, the initial $\hat{u}_2$ coefficient vector in the new basis is related to the coefficient vector $\hat{u}_1$ in the old basis by $\hat{u}_2 = M\hat{u}_1$. Now, the true solution u is approximated by $u(x,t,\xi) \approx \sum_{i=1}^{N_2} \hat{u}_2^i(x,t)\Psi_2^i(\xi)$, on the interval $[\tau_1, \tau_2]$. Since we only have $\hat{u}_2^i$ at time τ_1, we solve the propagation equation (1.6), which computes the $\hat{u}_2^i$ coefficients up to time τ_2.

At this point we repeat the procedure in the previous paragraph to generate a new set of empirical basis functions for the interval $[\tau_2, \tau_3]$, project the old coefficients onto the new basis, and solve the new propagation equation to obtain a solution up to time τ_3. This process is then repeated for every timestep until we reach the final integration time.

2.4. Convergence. Let $u_m^k = \sum_{i=1}^m \hat{u}^i(x,t)\Psi^i(\xi)$ be the approximation to the true solution on a time interval $[\tau_i, \tau_{i+1}]$ constructed through empirical chaos by sampling a discrete set of k trajectories in the space of random variables, $\{\xi_l\}_{l=1}^K$. Let $\bar{u}_m^k$ be the projection of the true solution u onto the span of the k trajectories. Note that the projection error at any of the ξ_l points will be zero. Then, from the triangle inequality we have

$$\left\| u - u_m^k \right\| \leq \left\| u - \bar{u}_m^k \right\| + \left\| \bar{u}_m^k - u_m^k \right\|,$$

in some appropriate norm. The error in the first term can be controlled by choosing $\{\xi_l\}_{l=1}^K$ as interpolation nodes and increasing K. This follows from standard interpolation theory, assuming sufficient regularity of the solution u. The error in the second term is due to POD, where instead of projecting u onto the full span of the trajectories, we project onto a m-dimensional subspace. This error can be controlled by increasing m.

2.5. Computational Cost. In order to fully solve the SPDE at an intermediate timestep $j+1$ (i.e. any timestep other than the first), we must perform the following operations:

(1) Solve the deterministic PDE for each point in $\{\xi_l\}_{l=1}^K$. This operation is simply K times the cost of solving a single instance of the deterministic PDE. However, this is trivially parallelizable, which can dramatically reduce the computational time.
(2) Compute the POD of the T matrix by computing its SVD. The T matrix always has K columns, but the number of rows depends on the number of finite-difference grid points and is almost always much larger than K. Since we only need the right singular values, which is a $K \times K$ matrix, we can reduce the cost of the SVD operation by only computing the singular value matrix Σ, and the right singular value matrix V.
(3) Project the old $\hat{u}_j^i$ coefficients onto the new set of basis functions. This requires us to compute the M matrix, which will be $N_{j+1} \times N_j$ (where N_j and N_{j+1} are the number of empirical basis functions for the j-th and $(j+1)$-st timesteps, respectively). Each entry of the matrix is computed by calculating an inner product with a numerical quadrature method. We then multiply the M matrix by the old $\hat{u}_j^i$ coefficients.
(4) Evaluate the propagation equation (1.6). This will require computing expectations, which can be done with numerical quadrature. The cost of this operation depends on the number of basis functions and also how complicated the differential operator L is.
(5) Solve the propagation equation (1.6) to advance the solution to the next timestep. The cost of this operation is highly dependent on the differential operator L. In some cases, this is no harder than solving the deterministic PDE, and in others it is far more computationally expensive. In either case, it is a coupled system of N_{j+1} PDEs.

The key advantage of this approach is that we can control the number of empirical basis functions by adjusting the size of the timestep. gPC uses the same set of basis functions for every point in time, and for many practical SPDEs this means that achieving accuracy in the solution out to long integration times requires using a very large set of basis functions. In contrast, each set of empirical basis functions only needs to have a low projection error over its corresponding time interval, which can be quite short. This allows the empirical chaos expansion method to run in linear time as a function of the final integration time, i.e., doubling the final integration time doubles the computational cost of empirical chaos expansion.

2.6. Computing Solution Statistics. If there are n total timesteps then we will need to keep track of a set of n sets of basis functions and their associated coefficients in order to fully reconstruct the solution. In the end, we still have a fully analytic representation of the true solution u, and we can use it to compute any desired solution statistics. The actual computations will be more complex than those for gPC, due to the fact that the basis functions are not orthogonal, and there are multiple sets of basis functions (one for each timestep). Let $\{\Psi_j^i\}_{i=1}^{N_j}$ be the set of N_j basis functions for timestep j, and let $\{\hat{u}_j^i\}_{i=1}^{N_j}$ be the corresponding set of coefficients. If we want to compute the mean of the true solution, then we compute,

$$E\left[u(x,t,\xi)\right] = E\left[\sum\nolimits_{i=1}^{N_{k^*}} \hat{u}_{k^*}^i \Psi_{k^*}^i\right] = \sum\nolimits_{i=1}^{N_{k^*}} \hat{u}_{k^*}^i E\left[\Psi_{k^*}^i\right],$$

where k^* is the timestep that contains the time t, and $E\left[\Psi_{k^*}^i\right]$ can be computed using numerical quadrature. Other solution statistics can be computed in a similar manner.

3. One-Dimensional Wave Equation

Consider the SPDE,

$$u_t(x,t,\xi) = \xi u_x(x,t,\xi), \quad 0 \le x \le 2\pi, \quad t \ge 0, \tag{3.1}$$

$$u(x,0,\xi) = \cos(x), \tag{3.2}$$

with periodic boundary conditions, and ξ is uniform on $[-1,1]$. The exact solution is

$$u(x,t,\xi) = \cos(x - \xi t). \tag{3.3}$$

Note that over the space for this problem,

$$\langle f, g\rangle := \int_{-1}^{1} f(\xi)g(\xi)(1/2)d\xi, \qquad E\left[f\right] := \int_{-1}^{1} f(\xi)(1/2)d\xi.$$

This hyperbolic SPDE was analyzed in [**18**], and one of the findings was that for a fixed polynomial basis, the error scales linearly with final integration time, thus requiring the gPC expansion to continually add additional terms in order to achieve accurate long-term solutions.

3.1. Polynomial Chaos Expansion.

When using gPC, we choose the normalized Legendre polynomials as our basis functions. Let L^i be the i-th normalized Legendre polynomial and apply stochastic Galerkin as in Section 1.1, and expand the true solution as

$$u(x,t,\xi) = \sum\nolimits_{i=1}^{\infty} \hat{u}^i(x,t) L^i(\xi).$$

Substituting into (3.1), multiplying by a test function $L^j(\xi)$, and taking the expectation yields

$$\begin{aligned}\hat{u}_t^j &= \sum\nolimits_{i=1}^{\infty} \hat{u}_t^i(x,t) E\left[L^i(\xi)L^j(\xi)\right] = E\left[\sum\nolimits_{i=1}^{\infty} \hat{u}_t^i(x,t)L^i(\xi)L^j(\xi)\right] \\ &= E\left[\xi \sum\nolimits_{i=1}^{\infty} \hat{u}_x^i(x,t)L^i(\xi)L^j(\xi)\right] = \sum\nolimits_{i=1}^{\infty} \hat{u}_x^i(x,t) E\left[\xi L^i(\xi)L^j(\xi)\right].\end{aligned}$$

Letting

$$A_{ji} = E\left[\xi L_j L_i\right], \quad \text{and} \quad \hat{u} = \begin{pmatrix} \hat{u}^1(x,t) \\ \hat{u}^2(x,t) \\ \vdots \end{pmatrix},$$

simplifies the previous system to

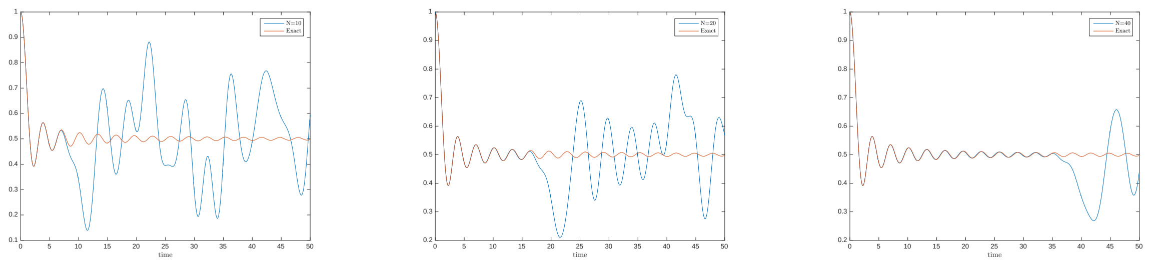
$$\hat{u}_t = A\hat{u}_x. \tag{3.4}$$

Since the initial condition is deterministic, $\hat{u}^1(x,0) = \cos(x)$, and $\hat{u}^i(x,0) = 0$ for $i > 1$. We truncate the infinite system to finite N and solve the resulting deterministic system of PDEs. The exact mean square expectation at $x = 0$ is given by

$$E\left[(u(0,t,\xi)^2\right] = \int_{-1}^{1} (\cos^2(\xi t))(1/2)d\xi = \frac{1}{2}\left(1 + \frac{\cos(t)\sin(t)}{t}\right). \tag{3.5}$$

See Figures 1a, 1b, 1c for a comparison of the exact value of the mean square expectation with the numerical value computed by the stochastic Galerkin method with different values of N. Past a certain point in time all of the PC expansions diverge from the exact value.

This is not an issue with the stochastic Galerkin method. In fact, even if we project the exact solution (3.3) onto the space of Legendre polynomials (see Figure 2) and compute its mean square expectation, we can see that more and more polynomial basis functions are needed to accurately project the exact solution.



(A) $d = 10$ (B) $d = 20$ (C) $d = 40$

FIGURE 1. Mean Square Expectation at $x = 0$ of solution to (3.1), computed using gPC with stochastic Galerkin using Legendre polynomials up to order d.

This raises another important concern with standard polynomial chaos techniques. If the initial condition is stochastic, then we may need a large number of polynomial functions *just to accurately project the initial condition*. For example, if we chose the initial condition to be the exact solution $\cos(x - \xi t)$ for a large value of t, then we would need a large number of Legendre polynomials even to accurately compute the solution over a short time interval.

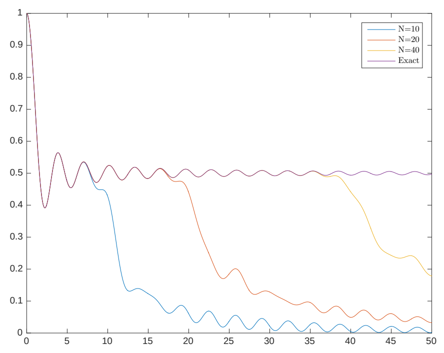


FIGURE 2. Mean Square Expectation at $x = 0$ for (3.1). Projection of the exact solution onto gPC basis functions for increasing polynomial order N.

3.2. Empirical Chaos Expansion. In order to apply an empirical chaos expansion, we must account for the fact that the basis functions might not be orthogonal. This results in the stochastic Galerkin system becoming slightly more complicated than the one for gPC. If we let $\{\Psi^i(\xi)\}_{i=1}^N$ be a set of possibly non-orthogonal basis functions, the exact solution to (3.1) can be approximated by

$$u(x,t,\xi) \approx \sum_{i=1}^{N} \hat{u}^i(x,t)\Psi^i(\xi). \tag{3.6}$$

Substituting this into (3.1) results in $\sum_{i=1}^{N} \hat{u}_t^i(x,t)\Psi^i(\xi) = \xi \sum_{i=1}^{N} \hat{u}_x^i(x,t)\Psi^i(\xi)$. Multiplying by a test function $\Psi^j(\xi)$, and taking an expectation over Ω yields

$$\begin{aligned} \sum_{i=1}^{N} \hat{u}_t^i(x,t) E\left[\Psi^i(\xi)\Psi^j(\xi)\right] &= E\left[\sum_{i=1}^{N} \hat{u}_t^i(x,t)\Psi^i(\xi)\Psi^j(\xi)\right] \\ &= E\left[\sum_{i=1}^{N} \hat{u}_x^i(x,t)\Psi^i(\xi)\Psi^j(\xi)\xi\right] = \sum_{i=1}^{N} \hat{u}_x^i(x,t) E\left[\Psi^i(\xi)\Psi^j(\xi)\xi\right]. \end{aligned} \tag{3.7}$$

Letting

$$M_{ji} = E\left[\Psi^i(\xi)\Psi^j(\xi)\right], \quad A_{ji} = E\left[\Psi^i(\xi)\Psi^j(\xi)\xi\right], \quad \text{and} \quad u = \begin{pmatrix} u^1(x,t) \\ u^2(x,t) \\ \vdots \\ u^N(x,t) \end{pmatrix},$$

then (3.7) can be expressed as

$$Mu_t = Au_x. \tag{3.8}$$

If we know the empirical basis functions Ψ^i and the initial value of $\hat{u}^i$, then the matrices M and A can both be computed, and (3.8) can be solved to obtain the coefficients at later times.

We follow the method in Section 2 to construct empirical basis functions over small time intervals by sampling trajectories and applying POD. Fixing ξ makes the PDE deterministic, which can be solved with existing numerical methods. We use a method of lines discretization to approximate the spatial derivative and apply a fourth-order Runge–Kutta method to perform the time integration (`ode45` in MATLAB). The POD is truncated, and the empirical basis functions $\{\Psi^i(\xi)\}$ are generated. We do not interpolate the basis functions, but instead use a composite trapezoidal quadrature rule to compute the expectations and inner products.

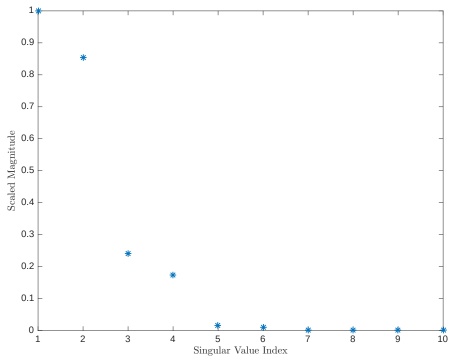
To determine an appropriate number of basis functions to use for a given timestep, we examine the singular values from POD in Figure 3a, scaled so that the first singular value is 1. There is a sharp drop-off in the singular values around the fifth singular value. In this case, we choose to truncate the expansion when the scaled singular values drops below 10^{-4}, which corresponds to the ninth singular value. The threshold is arbitrary and was determined empirically by truncating the expansion at varying values. This could possibly be made more precise by recalling that the squared Frobenius norm of the difference between a matrix and its truncated SVD is given by the sum of squares of the ignored singular values.

Figures 3b, 3c, and 3d show the result of applying empirical chaos expansion to the one-dimensional wave equation with 4, 5, and 9 basis functions, respectively. These used 120 Chebyshev nodes for ξ in the interval $[-1, 1]$ to compute trajectories for the empirical basis functions, with a timestep of size 1. The mean square expectation of the numerical solution was computed at $x = 0$ and compared to the true solution. Using only 4 or 5 empirical basis functions was inadequate for accuracy over the entire integration time, but 9 basis functions gave excellent agreement with the exact solution over the entire integration time, and adding additional basis functions does not appear to improve the accuracy.

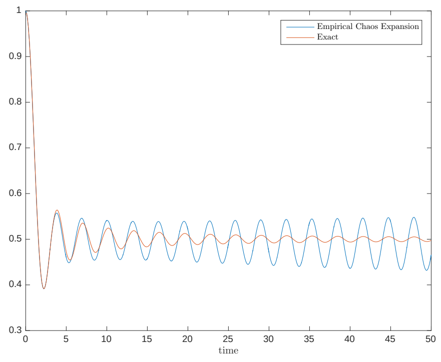
In general, choosing a larger timestep for the empirical chaos expansion results in growth of the required number of basis functions for each timestep. This is unsurprising, since we observe the same behavior with gPC; in order to accurately capture the solution behavior for long time periods, we need more and more basis functions. This presents an interesting optimization challenge for empirical chaos expansion, since a smaller basis results in a smaller stochastic Galerkin system, but a smaller timestep means that we must recompute the empirical bases more frequently. Recomputing the empirical basis involves sampling trajectories of the SPDE, computing the POD of the trajectories, constructing a new empirical basis, and projecting the data onto the new basis. Thus, a smaller timestep reduces the cost of solving the stochastic Galerkin system, but increases the cost of generating the empirical bases. For the model problem of this section, a timestep of size 4 appeared to be the optimal choice. Figure 4a shows the required number of basis functions for different choices of timesteps.

3.3. Running Time. As long as the number of empirical basis functions does not noticeably grow as the number of timesteps increases, we can expect the execution time for the empirical expansion method to scale linearly with the final integration time (see Figure 4b). When compared to the gPC method, we note the superlinear growth of the running time for gPC. However, for the timescales examined, gPC outperforms the empirical chaos method due to the fixed overhead cost of generating a new set of basis functions and performing the change of basis when applying an empirical chaos expansion.

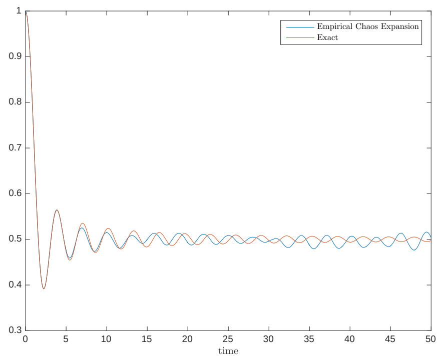
Since the Legendre polynomials and the A matrix in (3.4) can be precomputed for gPC, the running time to construct them is not included in the gPC running time. However, as the order gets larger, the running time to construct the A matrix is far larger than the actual time required to execute the gPC code, due to the $O(n^2)$ time required to compute all of the entries. In the simulations presented, the Legendre



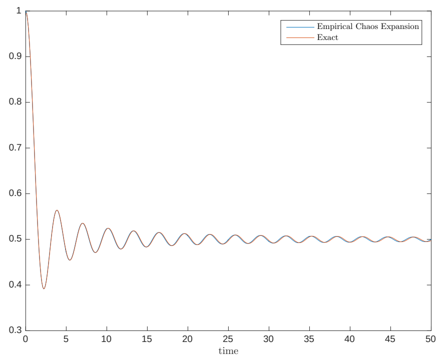
(A) Scaled singular values from POD for the first timestep.



(B) Maximum of 4 empirical basis functions on each timestep.



(C) Maximum of 5 empirical basis functions on each timestep.



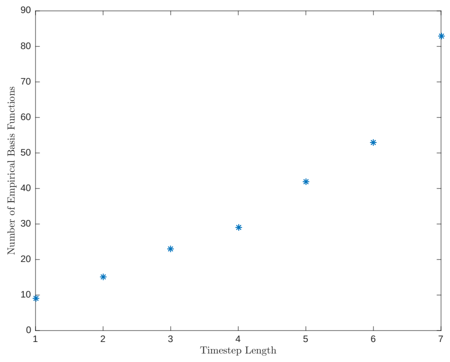
(D) Maximum of 9 empirical basis functions on each timestep.

FIGURE 3. Mean Square Expectation at $x = 0$ of the solution to (3.1), computed using empirical chaos expansion with stochastic Galerkin.

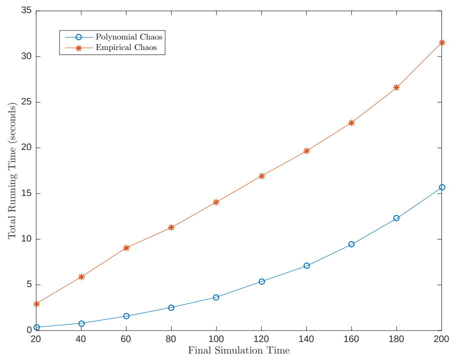
polynomials were generated symbolically and integrated exactly to avoid any error. In order to integrate out to time 200 accurately, we needed 220 basis functions, and the time required to compute the A matrix was around 10 hours. In contrast, the empirical chaos method requires no precomputation and instead computes its basis functions directly from the sampled trajectories.

3.4. Basis Function Evolution. Since a new set of basis functions is computed at every timestep in the empirical expansion, it is natural to examine how they change over time. In order to closely monitor their evolution, we set the timestep for the wave equation to 0.1, and examine the values of the basis functions over time. Figure 5a shows the basis function that corresponds to the largest singular value from POD for the first 5 timesteps (each of size 0.1).

The basis function evolves smoothly over time, but at singular value crossings it becomes associated with the second largest singular value. Figure 5b shows the magnitudes of the first and second singular values from POD for 100 timesteps, and there are multiple crossings. Figure 6 gives a three-dimensional view of the smoothness of the basis function evolution. Later, we will explore how to take advantage of this smooth evolution.

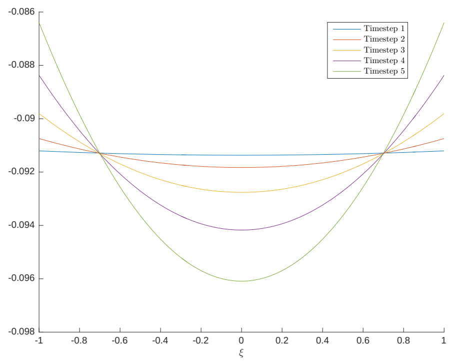


(A) Number of empirical basis functions required for an accurate solution for varying timesteps. Note the fast growth rate.

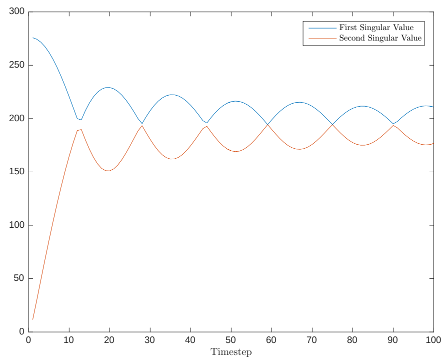


(B) Comparison of total running times for empirical chaos expansion and gPC with stochastic Galerkin to the same accuracy.

FIGURE 4. Computational scaling of empirical chaos expansion with stochastic Galerkin as a function of timestep and simulation length for the one-dimensional wave equation (3.1).



(A) Evolution of the first basis function from POD.



(B) Evolution of the magnitude of the first two singular values from POD.

FIGURE 5. Evolution of the basis functions and singular values from POD in the solution to (3.1) using empirical chaos expansion.

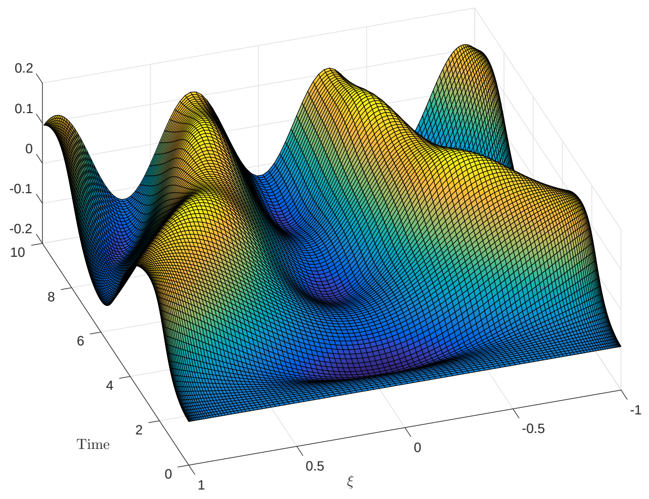


FIGURE 6. Evolution of the first basis function from POD for the first 100 timesteps in the solution to (3.1) using empirical chaos expansion.

4. Advection-Reaction Equation

The gPC method outperforms the empirical chaos expansion even though it uses a far larger polynomial basis for the wave equation. There are two reasons for this. First, the stochastic Galerkin PDE system was relatively straightforward to solve, even in high-dimensions. Additionally, the polynomial chaos expansions rely on precomputing the A matrix in (3.4) since it always uses the fixed set of Legendre polynomials as its orthogonal basis. This accounts for much of the efficiency observed in the previous section.

Consider the advection-reaction equation,

$$u_t(x,t,\xi) = \xi u_x(x,t,\xi) + f(x,u).$$

If $f(x,u)$ is nonlinear and not a simple polynomial such as u^2, the stochastic Galerkin system is more difficult to evaluate than the original system. If the basis is large, then the stochastic Galerkin system for standard polynomial chaos becomes computationally intractable, and the benefit of empirical chaos expansion manifests itself. By keeping the basis small, the stochastic Galerkin system remains relatively inexpensive even when the final integration time is large.

In the subsequent sections, we consider the model advection-reaction SPDE,

$$u_t(x,t,\xi) = \xi u_x(x,t,\xi) + 0.1\,|u|^{\frac{1}{2}}, \quad 0 \le x \le 2\pi, \quad t \ge 0, \tag{4.1}$$

$$u(x,0,\xi) = \cos(x) + \frac{3}{2}, \tag{4.2}$$

with periodic boundary conditions, and ξ is uniform on $[-1,1]$. Note that

$$\langle f,g\rangle := \int_{-1}^{1} f(\xi)g(\xi)(1/2)d\xi, \qquad E\,[f] := \int_{-1}^{1} f(\xi)(1/2)d\xi.$$

We do not have an analytic solution for this SPDE, so we run Monte Carlo simulations until they converge in order to obtain a reference solution to check the accuracy of our methods.

4.1. Polynomial Chaos Expansion. When using gPC, we choose the normalized Legendre polynomials as our basis functions. Let L^i be the i-th normalized Legendre polynomial and apply stochastic Galerkin as in Section 1.1. The details are similar to Section 3.1. Letting

$$A_{ji} = E\,[\xi L_j L_i]\,, \quad \hat{f}_j = 0.1E\left[\left|\sum_{i=1}^{\infty} \hat{u}_x^i(x,t)L^i(\xi)\right|^{\frac{1}{2}} L^j(\xi)\right], \quad \hat{u} = \begin{pmatrix} \hat{u}^1(x,t) \\ \hat{u}^2(x,t) \\ \vdots \end{pmatrix},$$

the gPC system can be written as

$$\hat{u}_t = A\hat{u}_x + \hat{f}. \tag{4.3}$$

Since the initial condition is deterministic,

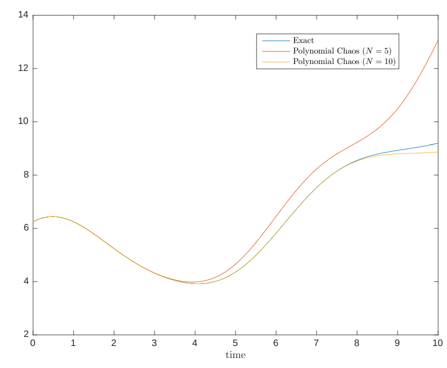
$$\hat{u}^1(x,0) = \cos(x) + \frac{3}{2}, \text{ and } \hat{u}^i(x,0) = 0 \text{ for } i > 1.$$

We can truncate the infinite system to finite N and solve the deterministic system of PDEs.

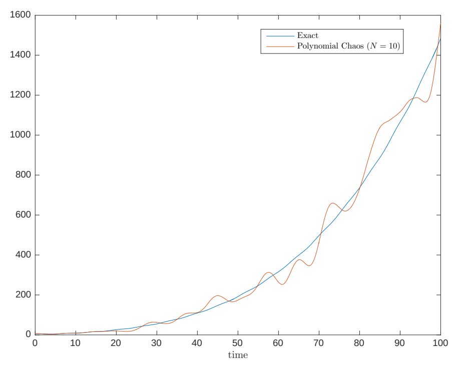
The stochastic Galerkin system in (4.3) is more difficult to solve than the original SPDE system in (4.1) due to the nonlinear $\hat{f}$ term. The expectations cannot be precomputed since they depend on $\hat{u}^i$ at each point in spacetime. We can compute $\hat{f}$ at each timestep with numerical quadrature, but it requires us to compute N expectations, where N is the number of basis functions. If we solve the stochastic Galerkin system using a method of lines discretization or some other finite-difference based approach, we will need to compute N expectations for every timestep of the numerical solver. This becomes very expensive as N grows. In order to compute the expectations in the $\hat{f}$ vector, we use a composite trapezoidal quadrature method that uses 300 Chebyshev nodes on the interval $[-1,1]$. The Legendre polynomials and the A matrix in (4.3) are precomputed using symbolic arithmetic to avoid any error.

To examine the solution accuracy, we consider the mean square expectation at $x=0$, computed from the numerical solution of the stochastic Galerkin system in (4.3). Figures 7a, 7b, 7c, and 7d show the numerical gPC results compared to the exact solution generated from 100,000 Monte Carlo simulations. For the short final time of 10 in Figure 7a, a small basis performs well, but for the longer final time of 100 in Figures 7b, 7c, and 7d, a much larger basis is required for accuracy. To numerically solve the differential equations, we use a method of lines discretization and a fourth-order Runge–Kutta integrator (`ode45` in MATLAB).

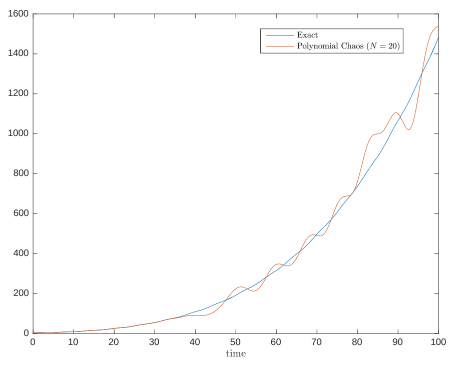
We include results up to polynomial order 40, but for higher orders, the stochastic Galerkin system became unstable. It is unclear if the instability was due to the numerical method used to solve the PDE system, or if the underlying system itself became ill-conditioned.



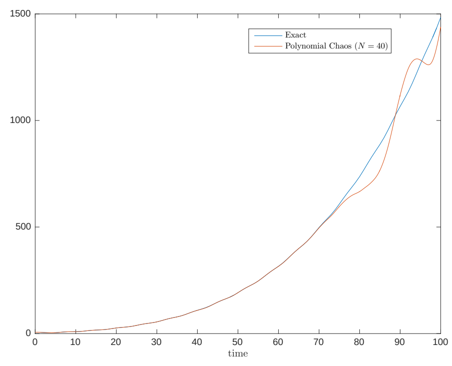
(A) Using N polynomial basis functions.



(B) Using Legendre polynomials up to order 10.



(C) Using Legendre polynomials up to order 20.



(D) Using Legendre polynomials up to order 40.

Figure 7. Mean square expectation $x = 0$ of solution to (4.1), computed using gPC with stochastic Galerkin. The exact solution was obtained from 100,000 Monte Carlo realizations.

4.2. Empirical Chaos Expansion. If we let $\{\Psi^i(\xi)\}_{i=1}^N$ be the set of empirical basis functions, we can perform a stochastic Galerkin method as in Section 1.1. The details are similar to Section 3.2. Letting

$$A_{ji} = E\left[\xi \Psi_j \Psi_i\right], \qquad M_{ji} = E\left[\Psi_j \Psi_i\right],$$

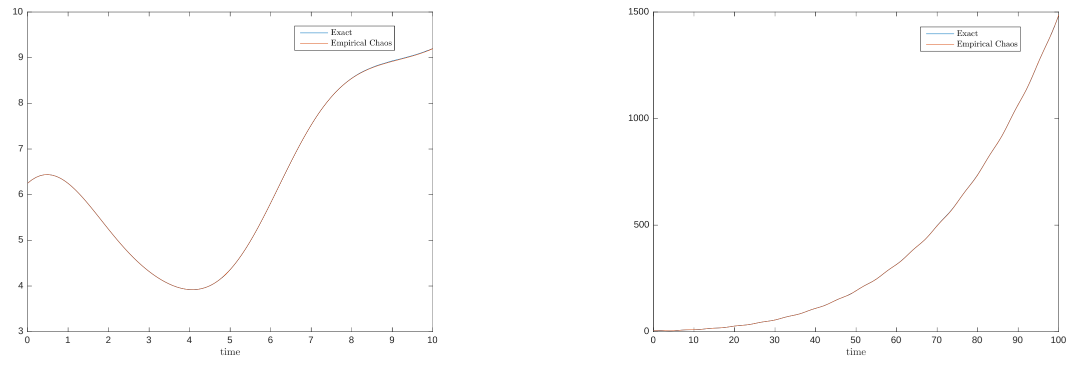
$$\hat{f}_j = 0.1 E\left[\left|\sum\nolimits_{i=1}^{N} \hat{u}_x^i(x,t)\Psi^i(\xi)\right|^{\frac{1}{2}} \Psi^j(\xi)\right], \qquad \hat{u} = \begin{pmatrix} \hat{u}^1(x,t) \\ \hat{u}^2(x,t) \\ \vdots \end{pmatrix},$$

the empirical chaos system can be expressed as

$$M\hat{u}_t = A\hat{u}_x + \hat{f}. \tag{4.4}$$

We follow the method in Section 2 to construct an empirical basis over small time intervals by sampling trajectories and applying POD, and use these to compute M, A, and $\hat{f}$ in (4.4). We use 300 Chebyshev nodes for ξ in the interval $[-1, 1]$. To fairly compare the computational cost with gPC, we compute the intermediate expectations of $\hat{f}$ using the same composite trapezoidal quadrature rule, and the same method of lines discretization and fourth-order Runge–Kutta integrator (`ode45` in MATLAB) to solve the stochastic Galerkin system.

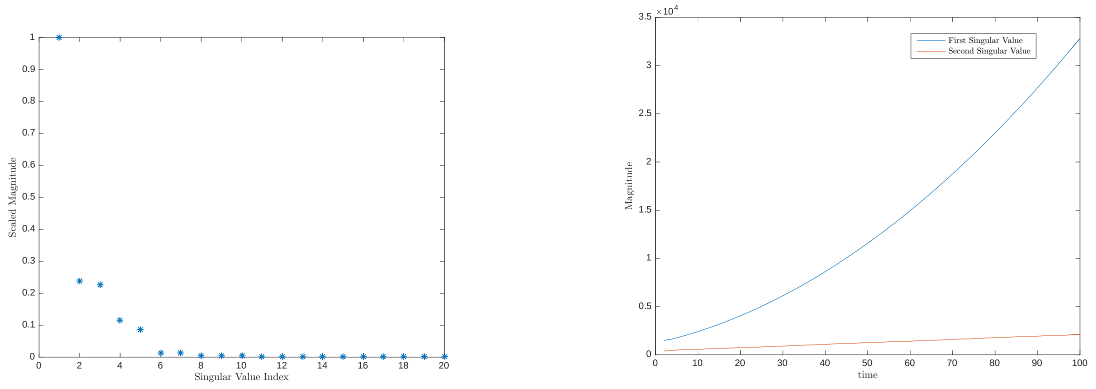
To examine the solution accuracy, we consider the mean square expectation at $x = 0$, computed from the numerical solution to the stochastic Galerkin system in (4.4). Figures 8a and 8b compare empirical chaos expansion with the exact solution from 100,000 Monte Carlo simulations. The sampling points were 300 Chebyshev nodes in the interval $[-1, 1]$.



(A) $t \in [0, 10]$. (B) $t \in [0, 100]$.

Figure 8. Mean square expectation at $x = 0$ of the solution to (4.1) computed using an empirical chaos expansion with stochastic Galerkin. The timestep size was 2, the basis functions were computed by solving the deterministic equation at 300 Chebyshev nodes on the interval $[-1, 1]$, and the maximum number of basis functions used at a single timestep was 33.

Just as before, we choose the number of empirical basis functions by examining the magnitude of the scaled singular values from POD. The overall number needed is higher for this SPDE than the previous ones, but there is still the same marked drop-off in the magnitude of the scaled singular values. Figure 9a shows the magnitude of the singular values from the POD that was computed at $t = 6$. There is a marked drop in the magnitude from the first to the second singular value, and by the sixth singular value the scaled magnitude is quite small.



(A) Scaled singular values from POD for the third timestep. (B) Evolution of the magnitude of the first two singular values from POD.

Figure 9. Singular values from POD in the solution to (4.1) using empirical chaos expansion.

The initial mean square expectation fluctuates due to the presence of the advection term ξu_x, but the reaction term $|u|^{\frac{1}{2}}$ dominates the behavior out at long times. This can be seen when comparing the mean square expectations in Figures 8a and 8b. In Figure 8a, the solution is integrated to final time 10, and there is a noticeable wave-like movement of the mean square expectation in time. In Figure 8b, however, the solution is integrated to final time 100, and though the wave-like motion is still present in the mean square expectation, it has been dominated by the total growth of the solution u due to the reaction term. The singular values from POD also reflect this behavior. In Figure 9b, we plot the magnitudes of the first two singular values that were generated at each timestep of the empirical chaos expansion solver. As the time grows, the first singular value becomes completely dominant, which reflects the fact that the reaction term dominates the behavior at later times.

4.3. Running Time. Since the number of empirical basis functions does not dramatically grow as the number of timesteps increases, the execution time for the empirical expansion method scales linearly with the final integration time (see Figure 10). Note the superlinear growth of the gPC running time as a function of the final integration time. Although gPC outperforms the empirical chaos expansion for smaller integration times, it is outperformed by the empirical chaos expansion for final integration times larger than about 34.

We do not include the cost to precompute the A matrix in (4.3), nor the cost to generate the Legendre polynomials for the polynomial chaos expansion. As mentioned previously, however, the entries of the $\hat{f}$ matrix cannot be precomputed, and the computational effort to compute these entries grows as $O(N^2)$, where N is the number of basis functions. As the final simulation time increases, we must increase the number of polynomial basis functions in order to accurately solve the system, and this causes the computational cost to grow superlinearly.

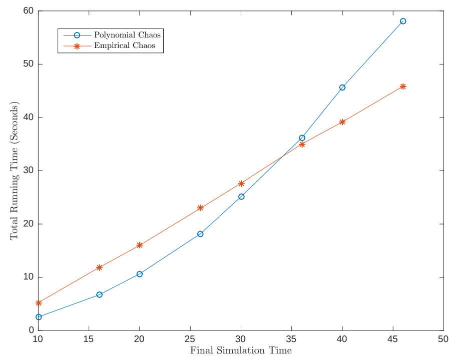
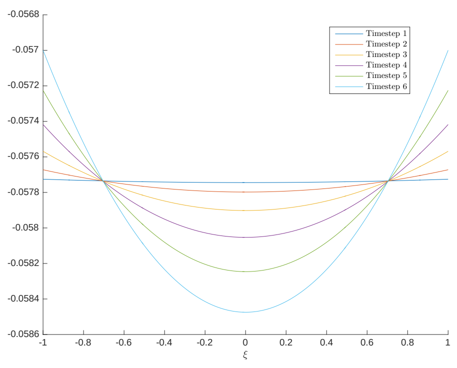
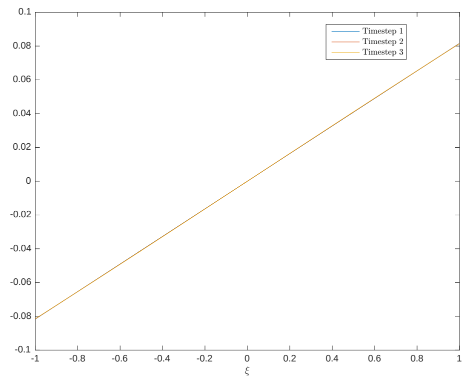


FIGURE 10. Comparison of total running times for solutions of (4.1) computed to the same accuracy using empirical chaos expansion and gPC with stochastic Galerkin.

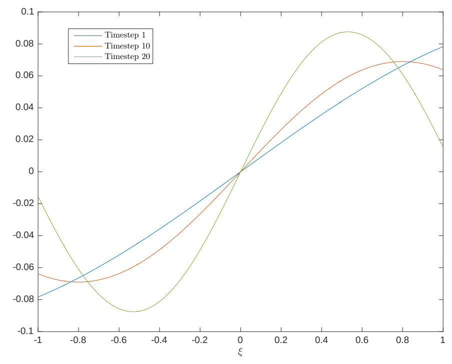
4.4. Basis Function Evolution. In Figure 11a, we examine the evolution of the basis function that corresponds to the largest singular value from POD for the advection-reaction equation (4.1) for the first 5 timesteps (each of size 0.1). The first basis function evolves smoothly over time, and since there are no crossings between the first and second singular value, it is straightforward to track its evolution. We also look at the second basis function evolution in Figures 11b and 11c. From Figure 11b, we see that the second basis function changes slowly, as is visually indistinguishable over the first 3 timesteps. The smooth evolution is more apparent in Figure 11c, which tracks it over the first 20 timesteps. The second and third singular values have multiple crossings in Figure 11d, which means that, depending on the time, the second basis function is associated with either the second or third singular value.



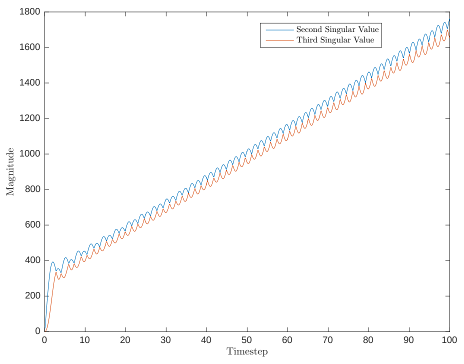
(A) Evolution of the first basis function from POD for the first five timesteps.



(B) Evolution of the second basis function from POD for the first three timesteps.



(C) Evolution of the second basis function from POD for the first 20 timesteps.



(D) Evolution of the magnitude of the second and third singular values from POD.

FIGURE 11. Evolution of the basis functions from POD in the solution to (4.1) using empirical chaos expansion.

5. Empirical Basis Evolution

The smooth evolution of the empirical basis functions derived from POD suggests that the empirical basis functions can be computed through an analytic approach that involves the dynamics of the SPDE. In gPC methods, we construct a basis that is orthogonal to the distribution of the random variables, project the exact solution onto this basis, and solve for the deterministic coefficients which are functions of spacetime. To examine how the dynamics of a SPDE influence the empirical basis functions for the random space, we consider an expansion in terms of a time-dependent basis for the random space,

$$u(x,t,\xi) \approx \sum\nolimits_{i=1}^{N} \hat{u}^i(x)\Psi^i(\xi,t), \tag{5.1}$$

and match this with the empirical chaos expansion (3.6) at time t_0, i.e., $\Psi^i(\xi,t_0) = \Psi^i(\xi)$ and $\hat{u}^i(x) = \hat{u}^i(x,t_0)$. The spatial coefficients are fixed and the random basis is time-dependent.

In order to evolve the basis functions in time, we consider a Galerkin method, this time integrating over the spatial component. We multiply by a test function $\hat{u}^j$, then integrate over the spatial domain, R. For the general model problem in (1.1), we obtain

$$\sum\nolimits_{i=1}^{N} \int_R \hat{u}^i(x)\hat{u}^j(x)\,dx\Psi^i(\xi,t) = \int_R L\left(\sum\nolimits_{i=1}^{N} \hat{u}^i(x)\Psi^i(\xi,t), x, t, \xi\right)\hat{u}^j(x)\,dx. \tag{5.2}$$

In practice, this system can usually be simplified, but it depends on the form of the differential operator L. This provides a DAE that describes the evolution of the basis functions of the random variable in time. The general approach we follow is to generate a set of empirical basis functions over a short time interval using POD, solve the resulting stochastic Galerkin system, and use the final values of the $\hat{u}^i(x,t)$ coefficients as the fixed spatial basis in (5.2). We use the empirical basis functions as the initial values of the $\Psi^i(\xi,t)$ coefficients, and then solve (5.2) over a short time interval and use the final values of the $\Psi^i(\xi,t)$ coefficients as the new empirical basis for the next stochastic Galerkin step.

5.1. One-Dimensional Wave Equation. If we substitute (5.1) into the one-dimensional wave equation (3.1), we get $\sum_{i=1}^N \hat{u}^i(x)\Psi^i_t(\xi,t) = \xi\sum_{i=1}^N \hat{u}^i_x(x)\Psi^i(\xi,t)$. Multiplying by a test function $\hat{u}^j$ and integrating over the spatial domain yields

$$\sum_{i=1}^N \int_0^{2\pi} \hat{u}^i(x)\hat{u}^j(x)\ dx\Psi^i_t(\xi,t) = \xi\sum_{i=1}^N \int_0^{2\pi} \hat{u}^i_x(x)\hat{u}^j(x)\ dx\Psi^i(\xi,t).$$

Letting

$$\Psi = \begin{pmatrix} \Psi^1 \\ \Psi^2 \\ \vdots \\ \Psi^N \end{pmatrix}, \quad \hat{u} = \begin{pmatrix} \hat{u}^1 \\ \hat{u}^2 \\ \vdots \\ \hat{u}^N \end{pmatrix}, \quad A_{ji} = \int_0^{2\pi} \hat{u}^i(x)\hat{u}^j(x)\ dx, \quad M_{ji} = \int_0^{2\pi} \hat{u}^i_x(x)\hat{u}^j(x)\ dx,$$

implies that

$$A\Psi_t(\xi,t) = \xi M\Psi(\xi,t), \tag{5.3}$$

which is a DAE that describes the evolution of basis functions in time. The general approach is to first generate a set of empirical basis functions up to a particular time t^*, using the method of Section 2. Then, the set of empirical basis functions is evolved in time by solving (5.3). In particular, if the matrix A is invertible, then the exact solution to (5.3) is

$$\Psi(\xi,t) = \exp\left[\xi A^{-1}M(t-t_0)\right]\Psi(\xi,t_0), \tag{5.4}$$

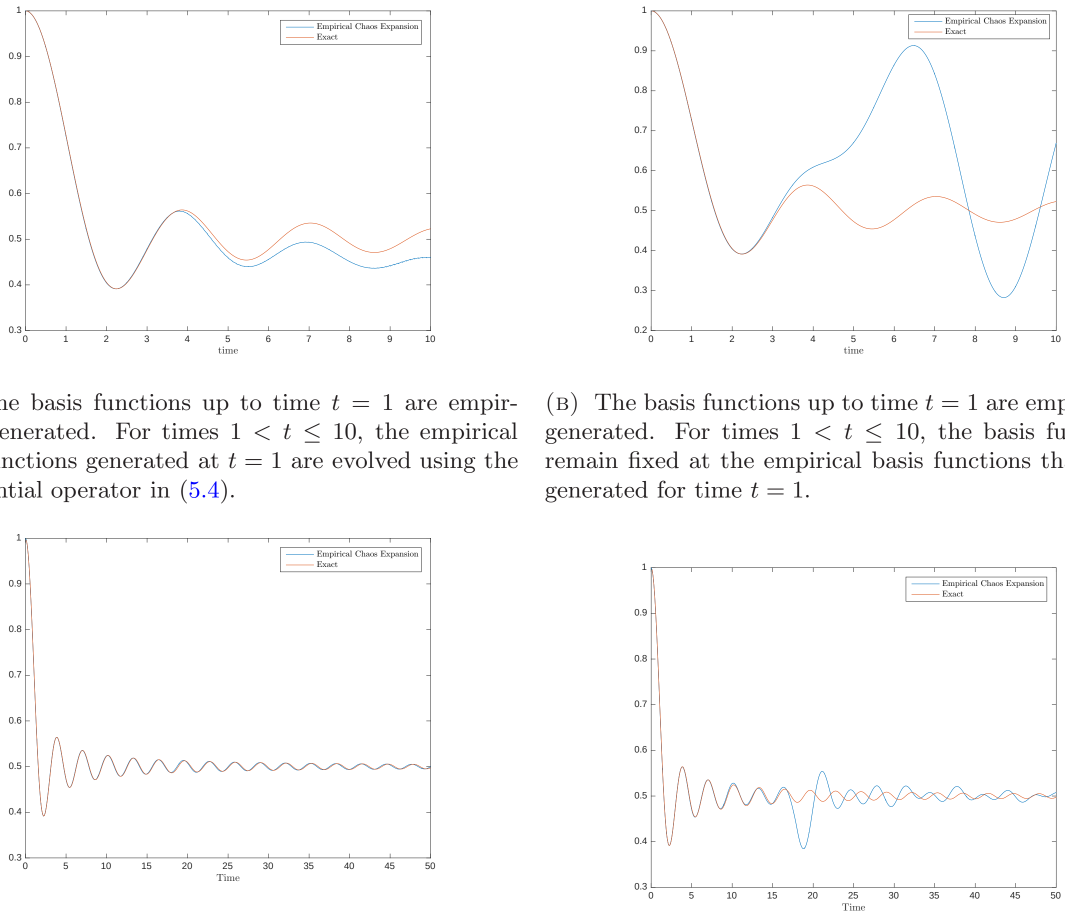
where exp is the matrix exponential operator. In practice, A is singular. To deal with this, we can decompose the full system into subsystems where the A matrix is nonsingular. To accomplish this, we choose the largest nonsingular square submatrix of A whose upper left corner is the $(1,1)$ entry of A. If that submatrix has lower right corner (k,k) where $k < n$, then we repeat this procedure by choosing another square matrix whose upper left corner is $(k+1,k+1)$, and so on. In practice, we calculate the condition number of the submatrices to determine if the matrix is close to singular. Then, each subsystem has a nonsingular A matrix, and the matrix exponential in (5.4) can be directly applied to update the basis functions.

Figure 12a illustrates this method applied to the one-dimensional wave equation (3.1). The basis functions up to time $t = 1$ are empirically generated using the standard method from Section 3. For times $1 < t \leq 10$, the empirical basis functions that were generated at $t = 1$ are evolved using the exponential operator from (5.4). The numerical solution closely matches the exact solution up to about $t = 4$. Past that time, the numerical solution begins to diverge from the exact solution, but still follows the general pattern. For comparison, in Figure 12b, the basis functions are empirically generated up to time $t = 1$, and then the basis functions are kept fixed for times $1 < t \leq 10$. When the basis functions are not evolved at all, the numerical solution matches the exact solution up to about $t = 3$, but wildly diverges from the exact solution beyond that, in much the same way as the standard polynomial chaos expansion.

The basis evolution operator involves a matrix exponential, and is potentially cheaper to compute than the trajectory sampling approach, which suggests the following general strategy:

(1) Compute sample trajectories and use POD to generate an empirical basis;
(2) Use the exponential basis evolution operator to update the basis functions for a few additional timesteps without resampling the trajectories;
(3) Repeat.

Although we do not resample the trajectories at each timestep, we need to use smaller timesteps when applying the matrix exponential operator. In the case of the one-dimensional wave, this results in a slight



(A) The basis functions up to time $t = 1$ are empirically generated. For times $1 < t \leq 10$, the empirical basis functions generated at $t = 1$ are evolved using the exponential operator in (5.4).

(B) The basis functions up to time $t = 1$ are empirically generated. For times $1 < t \leq 10$, the basis functions remain fixed at the empirical basis functions that were generated for time $t = 1$.

(C) The basis functions on even timesteps are empirically generated using POD. The basis functions on odd timesteps are evolved using the exponential operator in (5.4).

(D) The basis functions on even timesteps are empirically generated using POD. The basis functions on odd timesteps are not evolved from their previous values.

FIGURE 12. Mean square expectation at $x = 0$ of solution to (3.1), computed using empirical chaos expansion with stochastic Galerkin.

increase in the running time since the stochastic Galerkin system is already straightforward to solve. Figure 12c illustrates the result of the alternating timestep strategy, where the first timestep generates empirical basis functions by sampling trajectories, the second timestep evolves the basis functions using the exponential operator from (5.4), and so on. The solution retains the same accuracy as when we recomputed the basis at every timestep. In Figure 12d, we see that not using the exponential operator from (5.4) to update the basis functions on alternating timesteps causes the solution to quickly lose accuracy.

The procedure for the basis evolution approach is summarized in Algorithm 1, which couples empirical chaos expansion with basis evolution in a finite-differencing method.

Algorithm 1 Empirical Chaos with Basis Evolution

```
function ResampleBasis(timestep)    ▷ Return true if we should resample solutions to recompute the basis functions and return false if we should use the basis evolution operator. This can be a function that alternates between true and false, as in Figure 12c.
    if $i$ is odd then
        return True
    else
        return False
procedure EmpBasis
    $\xi[] \leftarrow$ vector with $n$ values from domain of random variable
    $\Delta t \leftarrow$ timestep size
    $n_t \leftarrow$ number of timesteps
    for $i = 1 \dots n_t$ do
        if ResampleBasis(i) then
            for $k = 1 \dots n$ do
                $s[k] \leftarrow$ Solution of (3.1) for fixed $\xi[k]$
            $T \leftarrow$ Matrix in (2.1) constructed from $s$
            $U, \Sigma, V^t \leftarrow \mathrm{svd}(T)$
            $k \leftarrow$ index of first scaled singular value smaller than $10^{-4}$
            $\Psi \leftarrow$ first $k-1$ columns of $V^t$
        else
            Apply operator in (5.4) to $\Psi$
        if i = 1 then
            $\hat{u} \leftarrow$ Projection of initial conditions onto $\Psi$
        else
            $\hat{u} \leftarrow$ Projection of $\hat{u}$ onto $\Psi$
        $\hat{u} \leftarrow$ Solution of (3.8)
```

6. Future Work

Since their inception, polynomial chaos techniques have been successfully applied to a number of numerical problems that arise in uncertainty quantification. This work introduces a method to generate a set of empirical basis functions that varies with time to take the place of the standard orthogonal polynomial basis that remains fixed for the entire integration time. We demonstrate its numerical accuracy and efficiency over long-term integrations with a few model problems. We also introduce a method to numerically evolve the empirical basis functions without needing to resample solutions of the original SPDE. We demonstrate that the running time of the empirical chaos method scales linearly with the final integration time. Thus, it has the potential to address one of the two principal issues with applying polynomial chaos techniques–solving problems out to long-term integrations is often inefficient due to the need to continually increase the number of polynomial basis functions.

This work does not, however, present a solution to the second and most fundamental issue–the curse of dimensionality. Namely, if the number of random variables is large then polynomial chaos techniques run slower than Monte Carlo methods. While empirical basis functions can be used in order to try to limit the size of such a basis, performing numerical integrations (which are a necessary component of the empirical chaos expansion) over a very high-dimensional space still poses significant challenges, both in terms of computational time and accuracy. We currently use sparse grid quadrature, which allows us to achieve accurate solutions over higher-dimensional spaces, but at the cost of sampling a large number of solutions of the original SPDE. Past a certain point, it becomes more practical to use Monte Carlo methods instead. Some techniques such as multi-element polynomial chaos have shown some promise when faced with larger numbers of random variables, and it is possible that using the basis decomposition employed by multi-element polynomial chaos methods coupled with empirical basis functions might allow higher-dimensional SPDEs to be solved more efficiently.

Another open question is to what degree the empirical chaos expansion algorithms can be made adaptive. Recall that the number of basis functions is selected by examining the scaled magnitude of the singular values from POD, and there are two values that we can adjust: (i) the length of each timestep; (ii) the number of sample trajectories of the SPDE. The timestep can be altered depending on the number of basis functions that we desire. In general, choosing a shorter timestep will result in the singular values from POD decaying faster, and thus will result in fewer empirical basis functions. The number of solutions to sample is a bit harder to determine. An approach that could be employed is to generate a set of empirical basis functions by sampling a fixed number of solutions to the SPDE, and then comparing the span of that basis to one that is generated by sampling a smaller number of solutions to the SPDE. If the difference in span is small then we would conclude that the number of solutions that we sampled is sufficient. If the difference in span is large, then we would conclude that the number of solutions that we sampled is insufficient, and proceed to sample additional solutions before repeating the process. It would be desirable to develop robust error estimates that could be used instead of generating two separate empirical bases.

The method of empirical basis expansion has the potential to be coupled with gPC expansions as well. We could use a set of N orthogonal polynomial functions and perform the same trajectory sampling that we do for an empirical chaos expansion. We could then project the trajectories onto the polynomial basis, and construct an empirical basis for the residuals of the trajectories. The empirical basis should be numerically orthogonal to the polynomial basis, which might allow the two systems to be propagated independently. Even if the two sets of basis functions need to be coupled, this work has demonstrated that such systems can be solved by including a mass matrix that turns the system of propagation PDEs into a DAE.

Recent work [**43**] has also shown that when gPC is applied to SPDEs with Hamiltonian structure, the resulting stochastic Galerkin system retains the Hamiltonian structure. Future work could attempt to determine whether the Hamiltonian structure is also retained when the original system is expanded using the empirical chaos method presented here.

We have shown the empirical chaos method with basis evolution to be robust, accurate, and efficient for the model problems presented. Further work is needed to explore its efficacy for a broader range of problems.

Acknowledgements

This research has been supported in part by NSF under grants DMS-1010687, CMMI-1029445, DMS-1065972, CMMI-1334759, DMS-1411792, DMS-1345013, DMS-1813635, and by AFOSR under grant FA9550-18-1-0288.

References

1. R Askey and J Wilson, *Some basic hypergeometric polynomials that generalize jacobi polynomials, vol. 319*, Mem. Amer. Math., Providence, RI (1985).
2. A Bensoussan and R Temam, *Equations stochastiques du type navier-stokes*, Journal of Functional Analysis **13** (1973), no. 2, 195–222.
3. Robert H Cameron and William T Martin, *Transformations of weiner integrals under translations*, Annals of Mathematics (1944), 386–396.
4. Anindya Chatterjee, *An introduction to the proper orthogonal decomposition*, Current science **78** (2000), no. 7, 808–817.
5. Alexandre Joel Chorin, *Gaussian fields and random flow*, Journal of Fluid Mechanics **63** (1974), no. 01, 21–32.
6. P-L Chow, *Stochastic partial differential equations in turbulence related problems*, NASA STI/Recon Technical Report A **79** (1978), 26135.
7. Giuseppe Da Prato and Arnaud Debussche, *Ergodicity for the 3d stochastic navier–stokes equations*, Journal de mathématiques pures et appliquées **82** (2003), no. 8, 877–947.
8. Guiseppe Da Prato, Arnaud Debussche, and Roger Temam, *Stochastic burgers' equation*, Nonlinear Differential Equations and Applications NoDEA **1** (1994), no. 4, 389–402.
9. Manas K Deb, Ivo M Babuška, and J Tinsley Oden, *Solution of stochastic partial differential equations using galerkin finite element techniques*, Computer Methods in Applied Mechanics and Engineering **190** (2001), no. 48, 6359–6372.
10. Alireza Doostan, Roger G Ghanem, and John Red-Horse, *Stochastic model reduction for chaos representations*, Computer Methods in Applied Mechanics and Engineering **196** (2007), no. 37, 3951–3966.
11. Alireza Doostan and Gianluca Iaccarino, *A least-squares approximation of partial differential equations with high-dimensional random inputs*, Journal of Computational Physics **228** (2009), no. 12, 4332–4345.
12. Oliver G Ernst, Antje Mugler, Hans-Jörg Starkloff, and Elisabeth Ullmann, *On the convergence of generalized polynomial chaos expansions*, ESAIM: Mathematical Modelling and Numerical Analysis **46** (2012), no. 02, 317–339.

13. Jasmine Foo and George Em Karniadakis, *Multi-element probabilistic collocation method in high dimensions*, Journal of Computational Physics **229** (2010), no. 5, 1536–1557.
14. Marc Gerritsma, Jan-Bart Van der Steen, Peter Vos, and George Karniadakis, *Time-dependent generalized polynomial chaos*, Journal of Computational Physics **229** (2010), no. 22, 8333–8363.
15. Roger Ghanem, *Ingredients for a general purpose stochastic finite elements implementation*, Computer Methods in Applied Mechanics and Engineering **168** (1999), no. 1, 19–34.
16. Roger Ghanem, George Saad, and Alireza Doostan, *Efficient solution of stochastic systems: application to the embankment dam problem*, Structural safety **29** (2007), no. 3, 238–251.
17. Roger G Ghanem and Pol D Spanos, *Stochastic finite elements: a spectral approach*, Courier Corporation, 2003.
18. David Gottlieb and Dongbin Xiu, *Galerkin method for wave equations with uncertain coefficients*, Commun. Comput. Phys **3** (2008), no. 2, 505–518.
19. Vincent Heuveline and Michael Schick, *A local time–dependent generalized polynomial chaos method for stochastic dynamical systems*, Preprint Series of the Engineering Mathematics and Computing Lab (2011), no. 04.
20. Thomas Y Hou, Wuan Luo, Boris Rozovskii, and Hao-Min Zhou, *Wiener chaos expansions and numerical solutions of randomly forced equations of fluid mechanics*, Journal of Computational Physics **216** (2006), no. 2, 687–706.
21. Michael B Isichenko, *Percolation, statistical topography, and transport in random media*, Reviews of modern physics **64** (1992), no. 4, 961.
22. Kari Karhunen, *Über lineare methoden in der wahrscheinlichkeitsrechnung*, vol. 37, Universitat Helsinki, 1947.
23. Gaetan Kerschen, Jean-claude Golinval, Alexander F Vakakis, and Lawrence A Bergman, *The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview*, Nonlinear dynamics **41** (2005), no. 1-3, 147–169.
24. VI Kliatskin, *Stochastic equations and waves in randomly inhomogeneous media*, Moscow Izdatel Nauka **1** (1980).
25. DD Kosambi, *Statistics in function space*, J. Indian Math. Soc **7** (1943), no. 1, 76–88.
26. OP Le Maıtre, HN Najm, RG Ghanem, and OM Knio, *Multi-resolution analysis of wiener-type uncertainty propagation schemes*, Journal of Computational Physics **197** (2004), no. 2, 502–531.
27. YC Liang, HP Lee, SP Lim, WZ Lin, KH Lee, and CG Wu, *Proper orthogonal decomposition and its applications part i: Theory*, Journal of Sound and vibration **252** (2002), no. 3, 527–544.
28. YC Liang, WZ Lin, HP Lee, SP Lim, KH Lee, and H Sun, *Proper orthogonal decomposition and its applications part ii: Model reduction for mems dynamical analysis*, Journal of Sound and Vibration **256** (2002), no. 3, 515–532.
29. Guang Lin, C-H Su, and George E Karniadakis, *Predicting shock dynamics in the presence of uncertainties*, Journal of Computational Physics **217** (2006), no. 1, 260–276.
30. Yiming Lou and Panagiotis D Christofides, *Estimation and control of surface roughness in thin film growth using kinetic monte-carlo models*, Chemical Engineering Science **58** (2003), no. 14, 3115–3129.
31. ———, *Feedback control of growth rate and surface roughness in thin film growth*, AIChE Journal **49** (2003), no. 8, 2099–2113.
32. ———, *Feedback control of surface roughness of gaas (0 0 1) thin films using kinetic monte carlo models*, Computers & chemical engineering **29** (2004), no. 1, 225–241.
33. D Lucor, D Xiu, and G Karniadakis, *Spectral representations of uncertainty in simulations: Algorithms and applications*, Proceedings of the International Conference on Spectral and High Order Methods (ICOSAHOM-01), Uppsala, Sweden, 2001.
34. R Mikulevicius and BL Rozovskii, *Stochastic navier–stokes equations for turbulent flows*, SIAM Journal on Mathematical Analysis **35** (2004), no. 5, 1250–1310.
35. Habib N Najm, *Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics*, Annual Review of Fluid Mechanics **41** (2009), 35–52.
36. Dong Ni and Panagiotis D Christofides, *Multivariable predictive control of thin film deposition using a stochastic pde model*, Industrial & Engineering Chemistry Research **44** (2005), no. 8, 2416–2427.
37. Anthony Nouy, *A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations*, Computer Methods in Applied Mechanics and Engineering **196** (2007), no. 45, 4521–4537.
38. ———, *Generalized spectral decomposition method for solving stochastic finite element equations: invariant subspace problem and dedicated algorithms*, Computer Methods in Applied Mechanics and Engineering **197** (2008), no. 51, 4718–4736.
39. Steven A Orszag and LR Bissonnette, *Dynamical properties of truncated wiener-hermite expansions*, Physics of Fluids (1958-1988) **10** (1967), no. 12, 2603–2613.
40. George C Papanicolaou, *Wave propagation in a one-dimensional random medium*, SIAM Journal on Applied Mathematics **21** (1971), no. 1, 13–18.
41. ———, *Diffusion in random media*, Surveys in applied mathematics, Springer, 1995, pp. 205–253.
42. E Pardouxt, *Stochastic partial differential equations and filtering of diffusion processes*, Stochastics **3** (1980), no. 1-4, 127–167.
43. Jose Miguel Pasini and Tuhin Sahai, *Polynomial chaos based uncertainty quantification in hamiltonian and chaotic systems*, Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, IEEE, 2013, pp. 1113–1118.
44. K Person, *On lines and planes of closest fit to system of points in space. philiosophical magazine, 2, 559-572*, 1901.
45. Matthew T Reagan, HN Najm, PP Pebay, OM Knio, and RG Ghanem, *Quantifying uncertainty in chemical systems modeling*, International journal of chemical kinetics **37** (2005), no. 6, 368–382.

46. Ya Sinai, *Burgers system driven by a periodic stochastic flow*, Itô's stochastic calculus and probability theory, Springer, Tokyo (1996), 347–353.
47. Ya G Sinai, *Two results concerning asymptotic behavior of solutions of the burgers equation with force*, Journal of statistical physics **64** (1991), no. 1-2, 1–12.
48. Pol D Spanos and Roger Ghanem, *Stochastic finite element expansion for random media*, Journal of engineering mechanics **115** (1989), no. 5, 1035–1053.
49. Manuel Villegas, F Augustin, Albert Gilg, A Hmaidi, and Utz Wever, *Application of the polynomial chaos expansion to the simulation of chemical reactors with uncertainties*, Mathematics and Computers in Simulation **82** (2012), no. 5, 805–817.
50. Xiaoliang Wan and George Em Karniadakis, *An adaptive multi-element generalized polynomial chaos method for stochastic differential equations*, Journal of Computational Physics **209** (2005), no. 2, 617–642.
51. ______, *Multi-element generalized polynomial chaos for arbitrary probability measures*, SIAM Journal on Scientific Computing **28** (2006), no. 3, 901–928.
52. E Weinan and Eric Vanden Eijnden, *Statistical theory for the stochastic burgers equation in the inviscid limit*, Commun. Pure Appl. Math. **53** (2000), 852–901.
53. E Weinan, K Khanin, A Mazel, and Ya Sinai, *Invariant measure for burgers equation with stochastic forcing*, Annals of Mathematics-Second Series **151** (2000), no. 3, 877–960.
54. E Weinan, Konstantin Khanin, Alexandre Mazel, and Yakov Sinai, *Probability distribution functions for the random forced burgers equation*, Physical Review Letters **78** (1997), no. 10, 1904.
55. Norbert Wiener, *The homogeneous chaos*, American Journal of Mathematics **60** (1938), no. 4, 897–936.
56. ______, *Nonlinear problems in random theory*, Nonlinear Problems in Random Theory, by Norbert Wiener, pp. 142. ISBN 0-262-73012-X. Cambridge, Massachusetts, USA: The MIT Press, August 1966.(Paper) **1** (1966).
57. Karen Willcox and Jaime Peraire, *Balanced model reduction via the proper orthogonal decomposition*, AIAA journal **40** (2002), no. 11, 2323–2330.
58. Dongbin Xiu, *Numerical methods for stochastic computations*, Princeton University Press, 2010.
59. Dongbin Xiu and George Em Karniadakis, *Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos*, Computer Methods in Applied Mechanics and Engineering **191** (2002), no. 43, 4927–4948.
60. ______, *The wiener–askey polynomial chaos for stochastic differential equations*, SIAM journal on scientific computing **24** (2002), no. 2, 619–644.
61. ______, *Modeling uncertainty in flow simulations via generalized polynomial chaos*, Journal of computational physics **187** (2003), no. 1, 137–167.
62. Dongbin Xiu, Didier Lucor, C-H Su, and George Em Karniadakis, *Stochastic modeling of flow-structure interactions using generalized polynomial chaos*, Journal of Fluids Engineering **124** (2002), no. 1, 51–59.
63. Thomas A Zang, Michael J Hemsch, Mark W Hilburger, Sean P Kenny, James M Luckring, Peiman Maghami, Sharon L Padula, and W Jefferson Stroud, *Needs and opportunities for uncertainty-based multidisciplinary design methods for aerospace vehicles*, (2002).