A Primer on Stochastic Galerkin Methods

Paul Constantine

March 22, 2007

1 Introduction

In the last five years, the scientific computing community has taken great interest in the so-called stochastic Galerkin schemes (SGS). These schemes are typically used to address the following type of problem: Suppose that I have a deterministic differential equation model for which I can find an approximate solution using standard numerical discretization techniques. Now suppose that, in an attempt to improve the model, I replace some deterministic model inputs with stochastic quantities that better represent my understanding of the problem. In other words, there may be some intrinsic uncertainty in the problem that I would like to account for in the model. One example of this may be an equation parameter that admits some measurement error, which we can model with a stochastic representation. Another example may be a randomly perturbed initial or boundary condition. Or perhaps the domain of the problem may have some “roughness” on the boundaries that we can represent with some stochastic quantities.

If the inputs to the model are now random, then the solution to the model will also be random. In effect, we have introduced a new stochastic dimension to the problem, or more accurately, we have accounted for the known uncertainties in the model using a probabilistic framework.

So how can I adapt the standard numerical techniques for solving the original deterministic problem to the new problem with randomness? Or, how can I alter the numerical techniques to *propagate* the uncertainty in the inputs and *quantify* the uncertainty in the solution? The stochastic Galerkin schemes offer one approach to these questions.

Before we move on, you may be asking why these schemes are called stochastic Galerkin. In a deterministic finite element framework, a standard Galerkin method projects the solution to the given differential equation onto a finite-dimensional basis. Then we can compute this “discretized” solution on our finite computers. The stochastic Galerkin schemes perform a similar projection in the random dimensions. In other words, they discretize the random dimensions to allow computation.

2 Mathematical Preliminaries

In this section, I briefly review some preliminary mathematical concepts needed to understand the stochastic Galerkin schemes. I assume that you have a basic familiarity with the tools used to solve deterministic numerical PDEs.

2.1 Randomness

Since we are introducing randomness into a deterministic system, it will help to review some basic concepts in probability. Modern probability theory is founded in the branch of analysis known as measure theory. I will not discuss any measure-theoretic concepts in this report, though I highly recommend gaining some familiarity with them for a deeper understanding and appreciation of probability theory. I have provided some excellent sources in the references section for further reading. If you already have a background in probability, then you can easily skip this section.

Definition *Continuous Random Variable* A *continuous random variable* $X(\omega)$ is a map from a sample space Ω to the real numbers, $X : \Omega \to \mathbb{R}$. It is called *continuous* because it can take on a continuum of values.

We use the notation $\mathbb{P}(X \leq x)$ to mean the *probability that X is less than or equal to x.* The next four definitions are very useful for characterizing the behavior of a continuous random variable.

Definition *Probability Density Funtion (pdf)* A *probability density function*, or pdf, of a continuous random variable X is a function $f_X : \mathbb{R} \to \mathbb{R}$ that has the following properties:

- $f_X(x) \geq 0$.
- f_X is piecewise continuous.
- $\int_{-\infty}^{\infty} f_X(x)\, dx = 1$.
- For $a, b \in \mathbb{R}$ with $a < b$, we have $\mathbb{P}(a < X < b) = \int_a^b f(x)\, dx$.

Definition *Cumulative Distribution Function* A *cumulative distribution function*, or cdf, of a continuous random variable X with pdf f_X is a function $F_X : \mathbb{R} \to [0, 1]$ defined as

$$F_X(x) = \int_{-\infty}^{x} f_X(u)\, du.$$

F_X has the following properties:

- F_X is non-decreasing.
- $\lim_{x\to\infty} F_X(x) = 1$, and $\lim_{x\to-\infty} F_X(x) = 0$.
- F_X is right-continuous.

Definition *Expectation* The *expectation* (or *mean*) of a random variable X with pdf f_X is given by

$$\mathbf{E}[X] \equiv \int_{-\infty}^{\infty} x f_X(x)\, dx.$$

Definition *Variance and Standard Deviation* The *variance* of a random variable X with pdf f_X is given by

$$\begin{aligned} \mathrm{Var}[X] &\equiv \int_{-\infty}^{\infty} (x - \mathbf{E}[X])^2 f_X(x)\, dx \\ &= \mathbf{E}[X^2] - (\mathbf{E}[X])^2. \end{aligned}$$

The *standard deviation* of X is defined as

$$\mathrm{Std}[X] \equiv \sqrt{\mathrm{Var}[X]}.$$

One very important result in our context concerns computing the statistics of functions of random variables. If X is a random variable, then for a function $g : \mathbb{R} \to \mathbb{R}$, $g(X)$ is also a random variable and

$$\mathbf{E}[g(X)] = \int_{-\infty}^{\infty} g(x) f_X(x)\, dx, \qquad \mathrm{Var}[g(X)] = \int_{-\infty}^{\infty} (g(x) - \mathbf{E}[g(X)])^2 f_X(x)\, dx.$$

The next definition describes a class of relatively well-behaved random variables. The random parameters we choose in engineering models are typically members of this class.

Definition *L_2 Random Variables* A random variable X is in the space L_2 if

$$\mathbf{E}[X^2] < \infty.$$

If $X \in L_2$, then it is sometimes called an L_2 *random variable*.

Equipped with this space, we can define convergence of a sequence of random variables in L_2.

Definition *L_2 Convergence* A sequence of random variables X_n, $n \in \mathbb{N}$ converges to X in L_2 if

$$\lim_{n\to\infty} \mathbf{E}\left[|X - X_n|^2\right] = 0.$$

In this case, we can also write

$$X_n \xrightarrow{L_2} X.$$

The next three definitions are important when describing a finite or countable collection of random variables.

Definition *Independence* Two random variables X_1 and X_2 are *independent* if

$$\mathbb{P}(X_1 \in A, X_2 \in B) = \mathbb{P}(X_1 \in A)\mathbb{P}(X_2 \in B)$$

where $\mathbb{P}(X_1 \in A, X_2 \in B)$ means the probability that X_1 is in the interval A *and* X_2 is in the interval B.

It is very important to note that if X_1 and X_2 are independent, then their *joint probability density function* $f_{X_1,X_2} : \mathbb{R}^2 \to \mathbb{R}$ is given by

$$f_{X_1,X_2}(x_1, x_2) = f_{X_1}(x_1) f_{X_2}(x_2).$$

Definition *Uncorrelated Random Variables* Two random variables X_1 and X_2 are *uncorrelated* if

$$\mathbf{E}[X_1 X_2] = \mathbf{E}[X_1]\mathbf{E}[X_2].$$

Note that if X_1 and X_2 are Gaussian random variables, then uncorrelated implies independence, but this is not true in general.

Definition *Orthogonal Random Variables* A finite collection of zero-mean random random variables X_i, $i = 1, \ldots, N$ is orthogonal if

$$\mathbf{E}[X_i X_j] = C\delta_{ij}$$

where C is a constant and δ_{ij} is the Kronecker delta.

It is easy to see that if a collection of random variables is uncorrelated with zero-mean, then it is also orthogonal. The next definition is useful when we encounter a random quantity that varies over some spatial domain.

Definition *Random Field* A continuous *random field* $\alpha(\mathbf{x}, \omega)$ is a collection of random variables indexed by the parameter $\mathbf{x} \in \mathcal{D}$, where $\mathcal{D} \subset \mathbb{R}^n$ is the spatial domain. The field $\alpha(\mathbf{x}, \omega)$ is often characterized by its expectation

$$\mathbf{E}[\alpha(\mathbf{x}, \omega)] \; = \; \int_\Omega \alpha(\mathbf{x}, \omega)\, d\omega \; \equiv \; \bar{\alpha}(\mathbf{x})$$

and its *covariance function* $C_\alpha : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.

In most engineering applications, n is 1, 2, or 3. For a complete introduction to the concepts in probability theory, see the references section.

2.2 Orthogonal Polynomials

Orthogonal polynomials play an important role in representing the random quantities in the stochastic models. In this section, I will briefly review some of their basic properties.

Let $P_n(x)$ be a polynomial of degree exactly n. A set of polynomials $\{P_n(x), n \in \mathbb{N}\}$ is called *orthogonal* if it satisfies the following orthgonality condition:

$$\int_{\mathcal{D}} P_n(x)P_m(x)W(x)\,dx = h_n\delta_{nm}, \qquad n, m \in \mathbb{N}$$

where $\mathcal{D}$ is the support of $\{P_n\}$ (possibly infinite), $W(x)$ is a specified weight function, h_n are non-zero constants, and δ_{nm} is the Kronecker delta. If $h_n = 1$, then the set is called orthonormal.

We can use the weight function $W(x)$ to define an inner product for two polynomial functions $f(x)$, $g(x)$ by

$$\langle f, g\rangle_W \equiv \int_D f(x)g(x)W(x)\,dx$$

Now observe that if the $W(x)$ is equal to a probability density function for some random variable X, and if $W(x)$ is also the weight function for some set of orthongal polynomials $\{P_n(x)\}$, then

$$\langle P_n(X), P_m(X)\rangle_W = h_n\delta_{nm}, \qquad n, m \in \mathbb{N}.$$

(Note that the arguments to the inner product are polynomial functions of the random variable.) It turns out that these strange coincident properties of the weight function hold for certain classes of orthgonal polynomials and random variables. This coincidence is extremely important for the stochastic Galerkin schemes. See the first appendix for a table with some of these relationships between classical orthogonal polynomials and continuous random variables.

Another important property of orthogonal polynomials is that they satisfy a three-term recurrence relation, which for a set $\{P_n(x)\}$ can be written

$$-xP_n(x) = A_nP_{n+1}(x) - (A_n + C_n)P_n(x) + C_nP_{n-1}(x), \quad n \geq 1$$

where A_n, $C_n \neq 0$ and $C_n/A_{n-1} > 0$. Thus by specifying P_{-1} and P_0, we can determine - or generate - all P_n if we know A_n and C_n.

We can build multi-dimensional orthogonal polynomials by taking products of one-dimensional orthogonal polynomials. For example, using an appropriate indexing function $k : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, the set of two-dimensional polynomials defined by

$$P_{k(n,m)}(x, y) = P_n(x)P_m(y), \qquad n, m \in \mathbb{N}$$

is orthogonal with respect to the weight function $W(x)W(y)$. Note again that if X and Y are independent random variables with joint density function $W(x)W(y)$, then the same orthogonality property holds for two-dimensional orthogonal polynomials of independent random variables. Following this method, we can construct orthogonal polynomials of arbitrary dimension.

2.3 Karhunen-Loeve Expansion

The Karhunen-Loeve expansion (KLE) has proved to be a very useful tool in representing the random input quantities in the stochastic models. Let $\alpha(\mathbf{x}, \omega)$ be a spatially varying random field over the spatial domain $\mathcal{D}$ with mean $\bar{\alpha}(\mathbf{x})$ and covariance function $C_\alpha(\mathbf{x}_1, \mathbf{x}_2)$. Then $\alpha(\mathbf{x}, \omega)$ can be represented as an infinite series

$$\alpha(\mathbf{x}, \omega) = \bar{\alpha}(\mathbf{x}) + \sum_{k=1}^{\infty} \sqrt{\lambda_k}\phi_k(\mathbf{x})\xi_k(\omega).$$

In the series above, λ_k and ϕ_k are the eigenvalues and eigenfunctions of the covariance function. In other words they solve the integral equations

$$\int_{\mathcal{D}} C_\alpha(\mathbf{x}_1, \mathbf{x}_2)\phi_k(\mathbf{x}_2)\,d\mathbf{x}_2 = \lambda_k\phi_k(\mathbf{x}_1).$$

These eigenpairs are known a priori for certain covariance functions. If they are not known beforehand for a given covariance function, they can be approximated numerically. The eigenvalues λ_k decay as k increases. The ξ_k in the series above are a collection of uncorrelated random variables with zero-mean and unit variance. The distributions of the ξ_k may be known or assumed beforehand, or they may be "fitted" from available data.

Obviously, we cannot execute real computations using an infinite series. Therefore we truncate the series after a finite number of terms, which gives an approximate representation of $\alpha(\mathbf{x}, \omega)$.

$$\alpha(\mathbf{x}, \omega) = \bar{\alpha}(\mathbf{x}) + \sum_{k=1}^{N} \sqrt{\lambda_k} \phi_k(\mathbf{x}) \xi_k(\omega).$$

The truncated KLE is known to be the finite representation with the minimal mean-square error over all such finite representations. Note that this series representation decouples the random dimensions from the spatial dimensions of $\alpha(\mathbf{x}, \omega)$.

2.4 Polynomial Chaos Expansion

If the covariance function of a random process $u(\mathbf{x}, \omega)$, $\mathbf{x} \in \mathcal{D}$ is not known – as is the case for the solution of a PDE with random inputs, then we can represent it using a the polynomial chaos expansion (PCE) given by

$$u(\mathbf{x}, \omega) = \sum_{k=0}^{\infty} u_k(\mathbf{x}) \Psi_k(\boldsymbol{\xi}).$$

In this case, $u_k(\mathbf{x})$ are deterministic coefficients. $\boldsymbol{\xi}$ is a vector of orthonormal random variables. And $\Psi_k(\boldsymbol{\xi})$ are multi-dimensional orthogonal polynomials with the following properties:

$$\langle \Psi_0 \rangle \equiv \mathbf{E}[\Psi_0] = 1, \qquad \langle \Psi_k \rangle = 0,\ k > 0, \qquad \langle \Psi_i \Psi_j \rangle = h_i \delta_{ij}.$$

By the Cameron-Martin theorem, the PCE of a random quantity converges in L_2, i.e.

$$\left\langle u(\mathbf{x}, \omega) - \sum_{k=0}^{\infty} u_k(\mathbf{x}) \Psi_k(\boldsymbol{\xi}) \right\rangle \xrightarrow{L_2} 0.$$

This convergence justifies a truncation to a finite number of terms for the sake of computation – similar in spirit to the truncated KLE,

$$u(\mathbf{x}, \omega) = \sum_{k=0}^{P} u_k(\mathbf{x}) \Psi_k(\boldsymbol{\xi}).$$

The value of P is determined by the number N of random variables – the length of $\boldsymbol{\xi}$ – and the highest degree of polynomial K used to represent u with the formula $P + 1 = (N + K)!/(N!K!)$. The value of N is the same as the truncation length of the truncated KLE, or more generally it is the number of uncorrelated random variables in the system. The value of K is typically chosen by some heuristic method. We will come back to this later.

It is worth noting before we move on that the KLE is a special case of the PCE with $K = 1$ and N random variables.

3 Stochastic Galerkin Technique

We now have the tools necessary to understand the stochastic Galerkin schemes. First, I will give an overview of the scheme using some general operators as in Xiu & Karniadakis (2002). Then I will give two simple examples – a 2D static Poisson equation and a 1D time dependent heat equation – complete with MATLAB code.

Consider the differential equation with random input parameters given by

$$\mathcal{L}(\mathbf{x}, t, \omega; u) = f(\mathbf{x}, t, \omega)$$

where $u = u(\mathbf{x}, t, \omega)$ is the solution and $f(\mathbf{x}, t, \omega)$ is the source term. $\mathcal{L}$ is a general differential operator that may contain spatial derivatives, time derivatives, and linear or non-linear terms. The ω denotes the dependence on some random input parameter, which may be equation parameters, boundary conditions, initial conditions, or even the domain.

The first step is to represent the solution with a truncated PCE.

$$u(\mathbf{x}, t, \omega) = \sum_{k=0}^{P} u_k(\mathbf{x}, t)\Psi_k(\boldsymbol{\xi}).$$

where $\boldsymbol{\xi}$ and $\{\Psi_k\}$ are chosen based on the characterization of the random input. Next we substitute this representation into the differential equation.

$$\mathcal{L}\left(\mathbf{x}, t, \omega; \sum_{k=0}^{P} u_k \Psi_k\right) = f(\mathbf{x}, t, \omega)$$

The choice of $\boldsymbol{\xi}$ and $\{\Psi_k\}$ defines a weight function for an inner product. Using this inner product, we take a Galerkin projection of the differential equation onto each basis polynomial Ψ_i.

$$\left\langle \mathcal{L}\left(\mathbf{x}, t, \omega; \sum_{k=0}^{P} u_k \Psi_k\right), \Psi_i \right\rangle = \langle f(\mathbf{x}, t, \omega), \Psi_i \rangle\,, \qquad i = 0, \ldots, P$$

The projection ensures that the error in the approximate solution is orthogonal to the space spanned by $\{\Psi_k\}$. By the orthogonality of $\{\Psi_k\}$, the "stochastic" differential equation reduces to a system of coupled deterministic differential equations for the coefficients of the truncated PCE. We can use any appropriate spatial and temporal discretization of the coefficients to solve this system.

Once we have the coefficients of the expansion, we can compute approximate statistics of the solution with the formulas

$$\begin{aligned}
\mathbf{E}[u] &= \mathbf{E}\left[\sum_{k=0}^{P} u_k \Psi_k\right] \\
&= u_0 \underbrace{\mathbf{E}[\Psi_0]}_{=1} + \sum_{k=1}^{P} u_k \underbrace{\mathbf{E}[\Psi_k]}_{=0} \\
&= u_0
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{Var}[u] &= \mathbf{E}[(u - \mathbf{E}[u])^2] \\
&= \mathbf{E}\left[\left(\left(\sum_{k=0}^{P} u_k \Psi_k\right) - u_0\right)^2\right] \\
&= \mathbf{E}\left[\left(\sum_{k=1}^{P} u_k \Psi_k\right)^2\right] \\
&= \sum_{k=1}^{P} u_k^2 \mathbf{E}[\Psi_k^2]; \qquad \text{(by orthogonality)}
\end{aligned}$$

and finally $\mathrm{Std}[u] = \sqrt{\mathrm{Var}[u]}$. We can also approximate the pdf of u by sampling from the distribution of $\boldsymbol{\xi}$ and plugging the samples into the PCE.

3.1 Example - 2D Poisson Equation

The first example is a Poisson equation in two spatial dimensions with one constant random parameter. Let $\mathcal{D} = \{(x,y) \; : \; -1 \leq x \leq 1, \; -1 \leq y \leq 1\}$ be the spatial domain of the problem and Ω the sample space. Then we seek a solution $u(x,y,\omega)$ that satisfies

$$\begin{aligned} \alpha(\omega)\left(u(x,y,\omega)_{xx} + u(x,y,\omega)_{yy}\right) &= 1 \qquad \text{on } \mathcal{D} \times \Omega \\ u(x,y,\omega) &= 0 \qquad \text{on } \partial\mathcal{D} \times \Omega \end{aligned}$$

where $\alpha(\omega)$ is a random variable distributed uniformly over the interval $[1,3]$.

First we express α in terms of a uniform random variable on $[-1,1]$, i.e.

$$\alpha = \xi + 2, \qquad \text{where } \xi \sim U[-1,1].$$

Next we represent u by a truncated PCE,

$$u(x,y,\omega) = \sum_{i=0}^{P} u_i(x,y)\Psi_i(\boldsymbol{\xi}(\omega)).$$

Since we have only one uniform random variable in the inputs, we choose $\boldsymbol{\xi} = \xi$. Once we choose the highest degree of polynomial K, then $P+1 = (K+1)!/K!$. To achieve exponential convergence in the coefficients u_i, we choose Ψ_i to be the 1D Legendre polynomials defined on the interval $[-1,1]$ (the range of ξ). Thus the weight function for the inner product is $W(x) = 1/2$ (the pdf of ξ). Substitute these expressions into the equation.

$$(\xi+2)\sum_{i=0}^{P}\left((u_i)_{xx} + (u_i)_{yy}\right)\Psi_i(\xi) = 1$$

We then perform a Galerkin projection by multiplying the equation by Ψ_k and taking the expectation (or inner product). Orthogonality of the $\{\Psi_i\}$ gives the system of equations

$$\begin{aligned} \sum_{i=0}^{P}\left((u_i)_{xx} + (u_i)_{yy}\right)\langle\Psi_k(\xi+2)\Psi_i\rangle &= \langle\Psi_k\rangle, \qquad \text{on } \mathcal{D} \\ u_k &= 0, \qquad \text{on } \partial\mathcal{D} \end{aligned}$$

for $k = 0,\ldots,P$. We can use central differencing to discretize in the spatial dimensions. Let $\mathbf{L}$ be the central differencing operator matrix representing the 2D five-point stencil in a uniform cartesian grid $\mathcal{D}_h$, with h the grid spacing. Then we can replace

$$\left((u_i)_{xx} + (u_i)_{yy}\right) \Longrightarrow \mathbf{L}\mathbf{u}_i$$

where $\mathbf{u}_i$ is a vector of the ith PCE coefficient discretized on the grid. Then the system becomes

$$\begin{aligned} \sum_{i=0}^{P}\langle\Psi_k(\xi+2)\Psi_i\rangle\mathbf{L}\mathbf{u}_i &= \langle\Psi_k\rangle\mathbf{e} \qquad \text{on } \mathcal{D}_h \\ \mathbf{u}_i &= 0 \qquad \text{on } \partial\mathcal{D}_h \end{aligned}$$

where $\mathbf{e}$ is a vector of 1s. We can write this very compactly using a Kronecker product. Define the $(P+1)\times(P+1)$ matrix $\mathbf{P}$ by

$$\mathbf{P}_{ij} = \langle\Psi_i(\xi+2)\Psi_j\rangle,$$

and define the vector $\mathbf{u} = (\mathbf{u}_0,\ldots,\mathbf{u}_P)^T$. Then we can solve for all the PCE coefficients at the grid points by solving the system

$$(\mathbf{P}\otimes\mathbf{L})\mathbf{u} = \mathbf{f}$$

where

$$\mathbf{f} \;=\; (\langle\Psi_0\rangle\mathbf{e}, \langle\Psi_1\rangle\mathbf{e}, \dots, \langle\Psi_P\rangle\mathbf{e})^T \;=\; (\mathbf{e}, \mathbf{0}, \dots, \mathbf{0})^T.$$

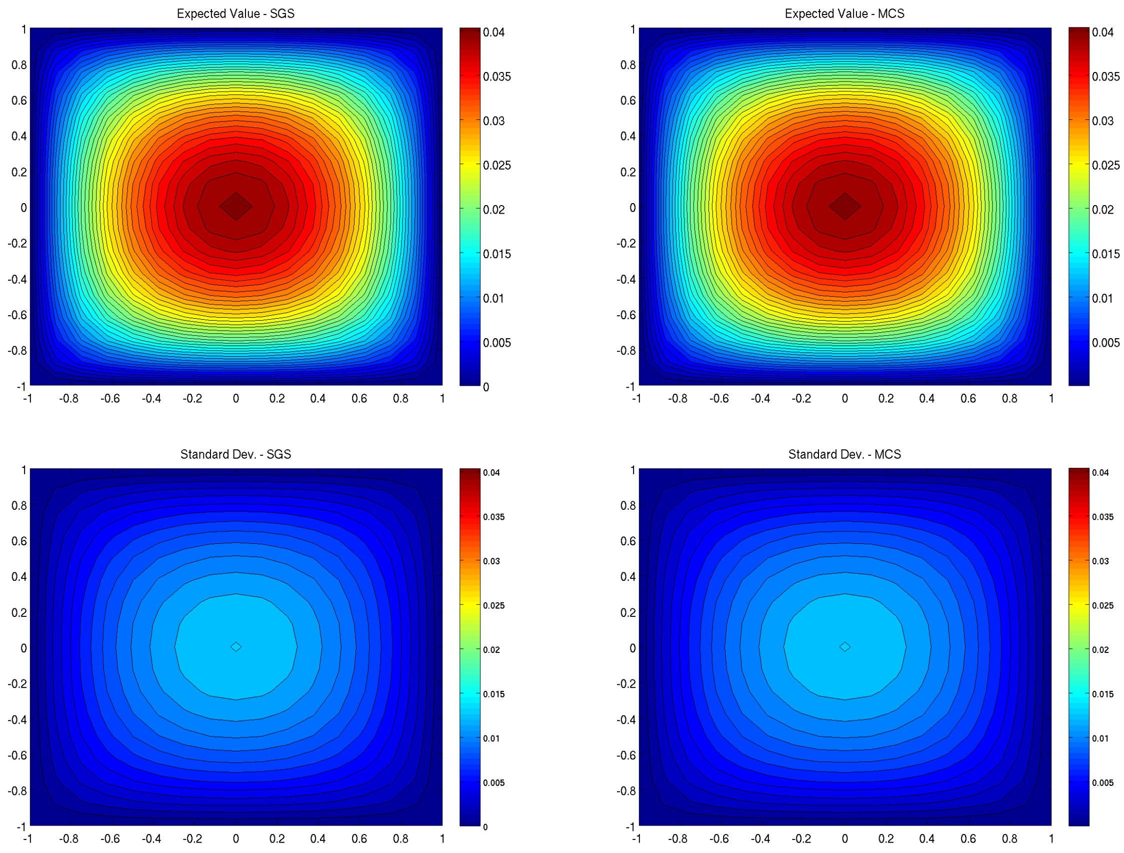
If the size of a comparable deterministic problem – one where α is a constant – is $M \times M$ ($\sqrt{M}$ grid points in each direction), then the size of the system to solve for the stochastic problem is $(P+1)M \times (P+1)M$.

I have written a MATLAB script to compute the expectation and standard deviation of the solution to this problem and compare them to a straightforward Monte Carlo method. You can easily modify the existing code to do convergence studies in P. Download the file `Poisson_2D.zip` from

`http://www.stanford.edu/~paulcon/SGS_Primer`

This code uses a toolbox I wrote for generating the orthogonal polynomials and computing the inner products symbolically. You will need MATLAB's symbolic toolbox. Make sure to read the README file before running the code.

The figure below compares the expected value and standard deviation of the solution using stochastic Galerkin and Monte Carlo.



Visually the results compare well, though you should use to code to check convergence of the solution.

3.2 Example - 1D Heat Equation

The second example is a transient heat conduction problem in one spatial dimension with a spatially varying random conductivity parameter. We seek a function $u(t, x, \omega)$ that satisfies

$$u_t(t, x, \omega) = (\alpha(x, \omega)u_x(t, x, \omega))_x + 1, \qquad x \in [-1, 1],\ t \in [0, T],\ \omega \in \Omega$$

with boundary conditions $u(t,-1,\omega) = u(t,1,\omega) = 0$ and initial condition $u(0,x,\omega) = 0$. The random field α is characterized by its mean and covariance function

$$\bar{\alpha} = 10, \qquad C_\alpha(x_1,x_2) = e^{-|x_1-x_2|}, \quad x_1,x_2 \in [-1,1]$$

One technical note: I chose $\bar{\alpha} = 10$ to ensure that the random fluctuations do not cause $\alpha \le 0$ over any part of the domain, which would result in an unstable backwards heat equation.

First we represent α by a truncated Karhunen-Loeve expansion.

$$\alpha(x,\omega) = \bar{\alpha} + \sum_{j=1}^{N} \sqrt{\lambda_j}\phi_j(x)\xi_j$$

where the eigenpairs $\{\lambda_j, \phi_j\}$ solve the integral equations

$$\int_{-1}^{1} e^{-|x_1-x_2|}\phi_j(x_2)\,dx_2 = \lambda_j\phi_j(x_1).$$

For this special case of the covariance function, we have explicit expressions for ϕ_j and λ_j. Let $\omega_{j_{\text{even}}}$ and $\omega_{j_{\text{odd}}}$ solve the equations

$$\begin{aligned} 1 - \omega_{j_{\text{even}}}\tan(\omega_{j_{\text{even}}}) &= 0 \\ \omega_{j_{\text{odd}}} + \tan(\omega_{j_{\text{odd}}}) &= 0 \end{aligned}$$

Then the even and odd indexed eigenfunctions are given by

$$\phi_{j_{\text{even}}}(x) = \frac{\cos(\omega_{j_{\text{even}}}x)}{\sqrt{1 + \frac{\sin(2\omega_{j_{\text{even}}})}{2\omega_{j_{\text{even}}}}}} \qquad \phi_{j_{\text{odd}}}(x) = \frac{\sin(\omega_{j_{\text{odd}}}x)}{\sqrt{1 - \frac{\sin(2\omega_{j_{\text{odd}}})}{2\omega_{j_{\text{odd}}}}}}$$

with corresponding eigenvalues

$$\lambda_{j_{\text{even}}} = \frac{2}{\omega_{j_{\text{even}}}^2 + 1} \qquad \lambda_{j_{\text{odd}}} = \frac{2}{\omega_{j_{\text{odd}}}^2 + 1}.$$

In most cases, the eigenfunctions and eigenvalues are approximated numerically. We choose $\boldsymbol{\xi} \equiv (\xi_1,\ldots,\xi_N)^T$ to be independent random variables uniformly distributed over the interval $[-1,1]$.

Next we represent u by a truncated PCE.

$$u(t,x,\omega) = \sum_{i=0}^{P} u_i(t,x)\Psi_i(\boldsymbol{\xi}(\omega)).$$

The value N is chosen by the truncation of the KLE of α. Once we choose the highest degree of polynomial K in the PCE, then $P = (N+K)!/(N!K!)$. To achieve exponential convergence in the coefficients $\{u_i\}$, we choose $\{\Psi_i\}$ to be the N-dimensional Legendre polynomials with support on the N-dimensional hypercube $[-1,1]^N$.

Before we substitute, let's make life a little easier by rewriting the original equation.

$$\begin{aligned} u_t &= (\alpha u_x)_x + 1 \\ &= \alpha u_{xx} + \alpha_x u_x + 1 \end{aligned}$$

Now we can substitute the PCE and KLE into this equation.

$$\sum_{i=0}^{P}(u_i)_t\Psi_i = \left(\bar{\alpha} + \sum_{j=1}^{N}\sqrt{\lambda_j}\phi_j\xi_j\right)\left(\sum_{i=0}^{P}(u_i)_{xx}\Psi_i\right) + \left(\sum_{j=1}^{N}\sqrt{\lambda_j}(\phi_j)_x\xi_j\right)\left(\sum_{i=0}^{P}(u_i)_x\Psi_i\right) + 1$$

Perform a Galerkin projection with Ψ_k.

$$\langle \Psi_k^2 \rangle (u_k)_t = \sum_{i=0}^{P} (u_i)_{xx} \left(\bar{\alpha} + \langle \Psi_k^2 \rangle \sum_{j=1}^{N} \langle \Psi_k \xi_j \Psi_i \rangle \sqrt{\lambda_j} \phi_j \right) + \sum_{i=0}^{P} (u_i)_x \left(\sum_{j=1}^{N} \langle \Psi_k \xi_j \Psi_i \rangle \sqrt{\lambda_j} (\phi_j)_x \right) + \langle \Psi_k \rangle$$

for $k = 0, \ldots, P$. We can discretize the PCE coefficients in time using Crank-Nicolson and in space with central differencing. Let $M+1$ be the number of spatial grid points. Define the following following $M-1 \times M-1$ matrices:

$$\mathbf{L}^{(2)} = \begin{bmatrix} -2 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -2 \end{bmatrix} \qquad \mathbf{L}^{(1)} = \begin{bmatrix} 0 & 1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & -1 & 0 \end{bmatrix}$$

are the second derivative and first derivative operators, respectively. Define the $P+1 \times P+1$ matrix $\mathbf{P}^{(j)}$ by the entries

$$\mathbf{P}_{ik}^{(j)} = \langle \Psi_k \xi_j \Psi_i \rangle$$

for $j = 1, \ldots, N$ and $i, k = 0, \ldots, P$. Also define the $P+1 \times P+1$ diagonal matrix $\mathbf{E}$ by

$$\mathbf{E} = \begin{bmatrix} \langle \Psi_0^2 \rangle & & & & \\ & \ddots & & & \\ & & \langle \Psi_j^2 \rangle & & \\ & & & \ddots & \\ & & & & \langle \Psi_P^2 \rangle \end{bmatrix}$$

Let x_m, $m = 0, \ldots, M$ be the grid points and define the $M-1 \times M-1$ diagonal matrices

$$\mathbf{D}_{\alpha}^{(j)} = \begin{bmatrix} \sqrt{\lambda_j}\phi_j(x_1) & & & & \\ & \ddots & & & \\ & & \sqrt{\lambda_j}\phi_j(x_m) & & \\ & & & \ddots & \\ & & & & \sqrt{\lambda_j}\phi_j(x_{M-1}) \end{bmatrix}$$

$$\mathbf{D}_{\partial\alpha}^{(j)} = \begin{bmatrix} \sqrt{\lambda_j}(\phi_j)_x(x_1) & & & & \\ & \ddots & & & \\ & & \sqrt{\lambda_j}(\phi_j)_x(x_m) & & \\ & & & \ddots & \\ & & & & \sqrt{\lambda_j}(\phi_j)_x(x_{M-1}) \end{bmatrix}$$

for $j = 1, \ldots, N$. Finally define the vectors of length $(P+1)(M-1)$

$$\mathbf{u}^n = (\mathbf{u}_0^n, \ldots, \mathbf{u}_P^n) \quad \text{where} \quad \mathbf{u}_i^n = (u_i^n(x_1), \ldots, u_i^n(x_{M-1}))^T,$$

and

$$\mathbf{f} \ = \ (\langle \Psi_0 \rangle \mathbf{e}, \langle \Psi_1 \rangle \mathbf{e}, \ldots, \langle \Psi_P \rangle \mathbf{e})^T \ = \ (\mathbf{e}, \mathbf{0}, \ldots, \mathbf{0})^T.$$

Then after manipulating the equation and utilizing the Kronecker product, we get the following system to solve at each timestep.

$$\left[\frac{1}{\Delta t} \mathbf{E} \otimes \mathbf{I}_{M-1} - \frac{1}{2\Delta x^2} \left(\sum_{j=1}^{N} \mathbf{P}^{(j)} \otimes \mathbf{D}_{\alpha}^{(j)} \right) \left(\mathbf{I}_{P+1} \otimes \mathbf{L}^{(2)} \right) - \frac{1}{4\Delta x} \left(\sum_{j=1}^{N} \mathbf{P}^{(j)} \otimes \mathbf{D}_{\partial\alpha}^{(j)} \right) \left(\mathbf{I}_{P+1} \otimes \mathbf{L}^{(1)} \right) \right] \mathbf{u}^{n+1}$$

$$= \left[\frac{1}{\Delta t} \mathbf{E} \otimes \mathbf{I}_{M-1} + \frac{1}{2\Delta x^2} \left(\sum_{j=1}^{N} \mathbf{P}^{(j)} \otimes \mathbf{D}_{\alpha}^{(j)} \right) \left(\mathbf{I}_{P+1} \otimes \mathbf{L}^{(2)} \right) + \frac{1}{4\Delta x} \left(\sum_{j=1}^{N} \mathbf{P}^{(j)} \otimes \mathbf{D}_{\partial\alpha}^{(j)} \right) \left(\mathbf{I}_{P+1} \otimes \mathbf{L}^{(1)} \right) \right] \mathbf{u}^n + \mathbf{f}$$

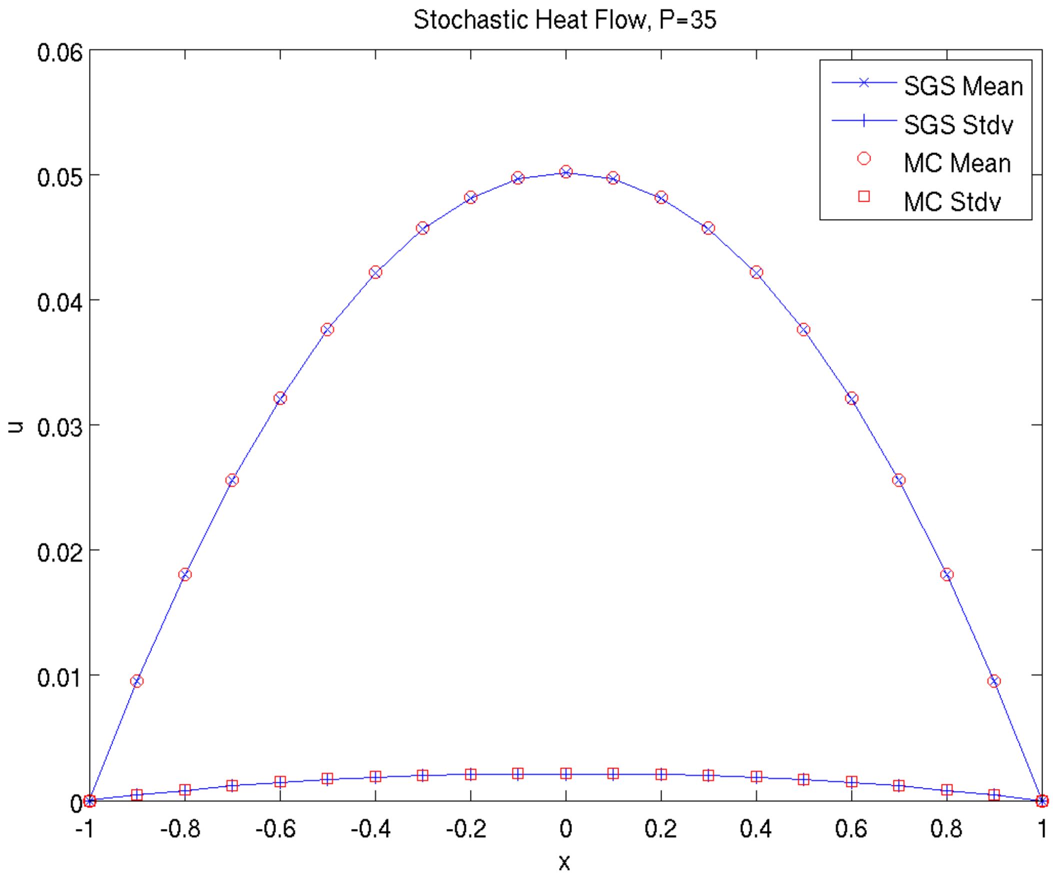
where $\mathbf{I}_{P+1}$ is a $P+1 \times P+1$ identity matrix and $\mathbf{I}_{M-1}$ is an $M-1 \times M-1$ identity matrix. Thus at each time step, we have to solve a linear system of size $(P+1)(M-1) \times (P+1)(M-1)$.

Once again, I have written a MATLAB script to compute the expectation and standard deviation of the solution and compare them to a straightforward Monte Carlo method. You can easily modify the code to perform convergence studies in P. Download the file `Heat_1D.zip` from

`http://www.stanford.edu/~paulcon/SGS_Primer`

This code uses a toolbox I wrote for generating the orthogonal polynomials and computing the inner products symbolically. You will need MATLAB's symbolic toolbox to run the code. Make sure to read the README file before running the code.

The figure below compares the expected value and standard deviation of the solution using stochastic Galerkin and Monte Carlo.



Visually the results compare well, though you should use to code to check convergence of the solution.

4 Summary and Future Work

This report was meant to give a practical introduction to stochastic Galerkin schemes with mathematical preliminaries and worked out examples with code. These schemes are used in the field of uncertainty quantification for propagating uncertainties associated with the inputs through a simulation and quantifying the resulting uncertainty in the output. The principle tools – the Karhunen-Loeve expansion and the polynomial chaos expansion – have strong theoretical foundations which justify this method.

I have not addressed the error analysis associated with this method. In fact, though much has been done in this area, researchers are still developing techniques to quantify the error associated with (1) truncating the PCE and (2) the independence (versus uncorrelated) assumption of the random variables in the KLE. This is a very active area of research.

One major drawback of this method is that the linear systems used to compute the PCE coefficients are typically much larger than the linear systems used to solve comparable deterministic problem. This is not terribly surprising since we are essentially introducing more dimensions into the problem. But researchers are currently searching for ways to either reduce the size of these systems or devise fast and efficient methods for solving them.

Another drawback of the stochastic Galerkin schemes from an engineering point of view is that simulation codes typically have to be rewritten to solve the coupled system of equations for the coefficients. Compare this to a straightfoward Monte Carlo approach – sampling from the distribution of the random inputs and running multiple deterministic simulations – which is much more expensive in terms of CPU time but does not require new codes. Researchers are exploring techniques such as importance sampling and collocation methods that could provide the accuracy of the stochastic Galerkin schemes using only a handful of deterministic simulations.

All these research activities are very exciting for those involved, since we believe that these types of problems with random inputs are ubiquitous. I hope this report has given you a taste for these problems. Please explore the references for further reading.

5 Appendices

The first appendix was taken from Xiu & Karniadakis (2002). The second appendix describes the PCETools MATLAB toolkit for generating orthogonal polynomials and computing inner products.

5.1 Orthogonal Polynomials and Random Variables

5.1.1 Hermite Polynomials and Gaussian r.v.'s

Parameters	
Support	$(-\infty, \infty)$
Recurrence	$H_{n+1} - 2xH_n + 2nH_{n-1} = 0$
Weight/pdf	$\frac{1}{\sqrt{2}} e^{-x^2/2}$

5.1.2 Laguerre Polynomials and Gamma r.v.'s

Parameters	$\alpha > -1$
Support	$[0, \infty)$
Recurrence	$(n+1)L_{n+1}^{(\alpha)} - (2n+\alpha+1-x)L_n^{(\alpha)} + (n+\alpha)L_{n-1}^{(\alpha)} = 0$
Weight/pdf	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$

5.1.3 Jacobi Polynomials and Beta r.v.'s

Parameters	$\alpha > -1$, $\beta > -1$
Support	$[-1, 1]$
Recurrence	$xP_n^{(\alpha,\beta)} = \frac{2(n+1)(n+\alpha+\beta+1)}{(2n+\alpha+\beta+1)(2n+\alpha+\beta+2)} P_{n+1}^{(\alpha,\beta)} + \frac{\beta^2-\alpha^2}{(2n+\alpha+\beta)(2n+\alpha+\beta+2)} P_n^{(\alpha,\beta)} + \frac{2(n+\alpha)(n+\beta)}{(2n+\alpha+\beta)(2n+\alpha+\beta+1)} P_{n-1}^{(\alpha,\beta)}$
Weight/pdf	$\frac{(1+x)^\beta (1-x)^\alpha}{2^{\alpha+\beta+1}\mathbf{B}(\alpha+1,\beta+1)}$

where

$$\mathbf{B}(\alpha+1,\beta+1) = \frac{\Gamma(\alpha+1)\Gamma(\beta+1)}{\Gamma(\alpha+\beta+2)}.$$

When $\alpha = \beta = 0$, the Jacobi polynomials reduce to the Legendre polynomials, and the weight function becomes a constant which corresponds to the distribution of the uniform random variables.

5.2 PCETools

This section documents how to use the PCETools MATLAB functions. These tools are for learning and experimenting. The are *not* efficient implementations for real scientific computing. You will need MATLAB's symbolic toolbox for these functions to work.

5.2.1 Generating Orthogonal Polynomials

To create a vector of length $n+1$ of d-dimensional Hermite polynomials up to order n, use

```
% creates the variables x1, x2, ..., xd
for i=1:d
    x(i) = sym(strcat('x',num2str(i)), 'real');
end
psi = pcepolymd('hermite', x, n);
```

The resulting vector contains the Hermite polynomials: `psi(i)` $= H_{i-1}(\mathbf{x})$. Similarly, to generate the other orthogonal polynomials up to order n, use

```
% creates the variables x1, x2, ..., xd
for i=1:d
    x(i) = sym(strcat('x',num2str(i)), 'real');
end

psi_legendre = pcepolymd('legendre', x, n);

alpha_laguerre = [alpha_1 alpha_2 ... alpha_d];
psi_laguerre = pcepolymd('laguerre', x, n, alpha_laguerre);

% support in each dimension, typically [-1,1]
a = [a_1 a_2 ... a_d];
b = [b_1 b_2 ... b_d];
alpha_jacobi = [alpha_1 alpha_2 ... alpha_d];
beta_jacobi = [beta_1 beta_2 ... beta_d];
psi_jacobi = pcepolymd('jacobi', x, n, a, b, alpha_jacobi, beta_jacobi);
```

5.2.2 Computing Inner Products

To analytically compute $\langle f, g\rangle$ against the various d-dimensional weighting functions, use the following commands.

```
% creates the variables x1, x2, ..., xd
for i=1:d
    x(i) = sym(strcat('x',num2str(i)), 'real');
end

pcepolymdIP('hermite', f, g, x);
pcepolymdIP('legendre', f, g, x);

alpha_laguerre = [alpha_1 alpha_2 ... alpha_d];
pcepolymdIP('laguerre', f, g, x, alpha_laguerre);

% support in each dimension, typically [-1,1]
a = [a_1 a_2 ... a_d];
b = [b_1 b_2 ... b_d];
alpha_jacobi = [alpha_1 alpha_2 ... alpha_d];
beta_jacobi = [beta_1 beta_2 ... beta_d];
pcepolymdIP('jacobi', f, g, x, a, b, alpha_jacobi, beta_jacobi);
```

6 References

6.1 Mathematical Fundamentals

- Rice, John A., *Mathematical Statistics and Data Analysis* – The first few chapters give an introduction to probability without measure theory.
- Williams, David, *Probability with Martingales* – A wonderful little textbook on probability theory with measure theory.
- Durrett, Rick, *Probability: Theory and Examples* – As the title suggests, with measure theory.
- Wiener, Norbert, *The Homogeneous Chaos*, American Journal of Mathematics, Oct. 1938 – The original paper, and a very tough read.
- Cameron, R.H. and Martin W.T., *The Orthogonal Development of Non-Linear Functionals in Series of Fourier-Hermite Functionals*, The Annals of Mathematics, Apr. 1947 – Convergence result for the PCE.

6.2 Stochastic Galerkin Methods

- Ghanem, R. and Spanos, P., *Stochastic Finite Elements: A Spectral Approach* – The first book on the subject. It's quite poorly written, but the material is essential.
- Xiu, D. and Karniadakis, G., *The Wiener-Askey Polynomial Chaos for Stochastic Differential Equations*, SIAM Journal of Scientific Computing, Vol. 24, No. 2. – Well-written for an overview. This is an essential read.
- Frauenfelder, P., Schwabb, C., Todor, A., *Finite Elements for Elliptic Problems with Stochastic Coefficients*, Computer Methods in Applied Mechanics and Engineering, 194 (2005) – More rigorous and detailed than the above papers.
- Babuska, I., Tempone, R., and Zouraris, G., *Galerkin Finite Element Approximations of Stochastic Elliptic Partial Differential Equations*, SIAM Journal of Numerical Analysis, Vol. 42, No. 2. – A slightly different approach.

This is a starting point. The references in each of these works contain many more important papers and applied examples. If you have more questions, you can send me an email at *paul.constantine@stanford.edu.*