STOCHASTIC SAMPLING METHODS

APPROXIMATING QUANTITIES OF INTEREST USING SAMPLING METHODS

- Recall that quantities of interest often require the evaluation of stochastic integrals of functions of the solutions
- These integrals usually have to be approximated using quadrature rules, i.e.,

$$\int_\Gamma G\big(u(\mathbf{x},\vec{y});\mathbf{x},\vec{y}\big)\rho(\vec{y})\,d\vec{y} \approx \sum_{q=1}^{Q} w_q G\big(u(\mathbf{x},\vec{y}_q);\mathbf{x},\vec{y}_q\big)$$

  or

$$\int_\Gamma G\big(u(\mathbf{x},\vec{y});\mathbf{x},\vec{y}\big)\rho(\vec{y})\,d\vec{y} \approx \sum_{q=1}^{Q} w_q \rho(y_q) G\big(u(\mathbf{x},\vec{y}_q);\mathbf{x},\vec{y}_q\big)$$

- To use such a rule, one needs to know the solution $u(\mathbf{x},\vec{y})$ of the SPDE at each of the quadrature points $\vec{y}_q$, $q=1,\ldots,Q$, in the probabilistic domain Γ
  - for this purpose, one can use a stochastic Galerkin method to obtain an approximation to the the solution $u(\mathbf{x},\vec{y})$ and then evaluate that approximation at the quadrature points

- However, once a quadrature rule is chosen to approximate a quantity of interest,

    - i.e., once the quadrature points $\{\vec{y}_q\}_{q=1}^{Q}$ are known

  the simplest and most direct means of determining $u(\mathbf{x}, \vec{y}_q)$ is to simply solve the PDE Q times, once for each quadrature point $\vec{y}_q$

- This approach is referred to as the stochastic sampling method (SSM) for SPDEs and for quantities of interest that depend on the solutions of SPDEs

- We have already encountered two SSMs

  – we have seen that SGMs based on Lagrange interpolating polynomials reduce to SSMs

  – we have also seen that non-intrusive polynomial chaos methods are essentially SSMs

    - although one does need the additional step of explicitly constructing the non-intrusive polynomial chaos approximation

- In an SSM, to determine an approximation to a quantity of interest,
  - one chooses a quadrature rule for the probabilistic integrals, i.e.,
    - one chooses quadrature weights and points $\{w_q, \vec{y}_q\}_{q=1}^{Q}$
  - one chooses a finite element method, (i.e., a finite element space and a basis $\{\phi_j\}_{j=1}^{J}$ for that space) and, for each q, one defines the finite element approximation of the solution at the quadrature points by

$$u_q(\mathbf{x}) = \sum_{j=1}^{J} b_{j,q}\phi_j(\mathbf{x}) \qquad \text{for } q = 1, \ldots, Q$$

  - then, to determine $b_{j,q}$ for $j = 1, \ldots, J$ and $q = 1 \ldots, Q$, one separately, and if desired, in parallel, solves the Q deterministic problems: for $q = 1, \ldots, Q$,

$$\int_{\mathcal{D}} S\Big(\sum_{j=1}^{J} b_{j,q}\phi_j, \vec{y}_q\Big) T(\phi_{j'})\, d\mathbf{x} = \int_{\mathcal{D}} \phi_{j'} f(\vec{y}_q)\, d\mathbf{x} \qquad \text{for } j' = 1, \ldots, J$$

- each of these can be discretized using a finite element method

  $\Longrightarrow$ one can use legacy codes as black boxes

  $\Longrightarrow$ i.e., without changing a single line of code

  $\Longrightarrow$ i.e., one just uses the legacy code Q times

– and finally, one just substitutes $u_q(\mathbf{x})$ wherever $u(\mathbf{x};\vec{y}_q)$ is needed into the quadrature rule approximation of a quantity of interest

- The cost of determining an approximation to a quantity of interest using the SSM approach is dominated by

  – the cost to determine Q finite element solutions, each of size J

- This should be compared to the cost of using general SGM approaches for the same purpose that are dominated by

  – the cost needed to determine the solution of
  a single system of size JK

- Which approach wins, i.e., which one yields a desired accuracy in the statistics of quantities of interest for the lowest computational cost, depends on
  - the value of Q, the number of quadrature points in SSM approaches
  - the value of K, the number of probabilistic terms in the SGM approximation to the solution
  - the cost of solving the systems of discrete equations encountered
    - for nonlinear problems and time dependent problems, one may have to solve such systems many times
  - many implementation issues

- Of course, such comparisons do not factor in the relative programming cost for implementing the different approaches
  - SSM approaches allow for the easy use of legacy codes
  - general SGM approaches do not allow for this

- In most cases, and certainly due to some recent developments, SSMs win over SGMs
  - which is why polynomial chaos people are now doing non-intrusive polynomial chaos which is, as we have seen, practically a SSM
- Of course, there are many ways to sample points in parameter space other than at the quadrature points for some integration rule
  - so, we now take a more general view of SSMs

STOCHASTIC SAMPLING METHODS ARE STOCHASTIC GALERKIN METHODS

- From the previous discussions, it seems that we could have introduced stochastic sampling methods as a special case of stochastic Galerkin methods
  - in fact,

every stochastic sampling method
is a stochastic Galerkin method using
Lagrange interpolating polynomials
based on the sample points
and quadrature rules also based on the sample points

- However, stochastic sampling methods are easier to understand through the straightforward approach we have just taken
  - the straightforward approach also avoids difficult questions about the relations of the cardinality of the set of sample points and the construction of interpolating polynomials

SURROGATE APPROXIMATIONS AND STOCHASTIC SAMPLING METHODS

- Stochastic sampling methods (SSMs) for solving stochastic PDEs are based on
  - first determining a sample set of values $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ of the vector of random parameters $\vec{y} \in \Gamma \subset \mathbb{R}^N$
  - then determining N_{sample} (approximate) solutions $\{u(\mathbf{x};\vec{y}_s)\}_{s=1}^{N_{sample}}$ of the PDE via, e.g., a finite element method

Evaluating quantities of interest within the SSM framework

- If we want to evaluate quantities of interest that involve integrals over the parameter set Γ using a Q-point quadrature rule involving the quadrature points $\{\vec{y}_q\}_{q=1}^{Q} \subset \overline{\Gamma}$ and quadrature weights $\{w_q\}_{q=1}^{Q}$
  - it is then natural to choose the set of sample points $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ that are used to solve the PDE N_{sample} times to be the same as the set of quadrature points $\{\vec{y}_q\}_{q=1}^{Q}$ that are used to approximate the quantities of interest

- Alternately, we could choose $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ to be different (and presumably coarser) than the quadrature points $\{\vec{y}_q\}_{q=1}^{Q}$
  - one would then use the sample points $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ to build a surrogate or response surface $u_{surrogate}(\mathbf{x}, y)$ for the solution $u(\mathbf{x}, y)$
  - surrogates/response surfaces for the solution $u(\mathbf{x}, \vec{y})$ are (usually polynomial) functions of, in our case, the random parameters $\vec{y}$

– in fact, they are simply representations, e.g., in terms of Lagrange interpolation polynomials, of the approximate solution in terms of the parameter vector $\vec{y}$

– it is usually more efficient to build a surrogate/response surface directly for the integrand $G\big(u(\mathbf{x},\vec{y});\mathbf{x},\vec{y}\big)$ of the desired quantity of interest

- one solves for an approximation $u_s(\mathbf{x})$ to the solution $u(\mathbf{x},\vec{y}_s)$ of the PDE for the sample parameter points $\vec{y}_s$, $s = 1,\ldots,N_{sample}$

- one then evaluates the approximations to the integrand

$$G_s(\mathbf{x}) = G\big(u_s(\mathbf{x});\mathbf{x},\vec{y}_s\big) \qquad \text{for } s = 1,\ldots,N_{sample}$$

- from these samplings of G at the sample points $\vec{y}_s$, one builds a surrogate $G_{surrogate}(\mathbf{x},\vec{y})$

– once a surrogate/response surface is built, it can be used to evaluate the integrand at the quadrature points $\{\vec{y}_q\}_{q=1}^{Q}$

- To illustrate the different approaches, within the SSM framework, for computing approximations of quantities of interest, consider a quantity of the form

$$\mathcal{J}(u) = \int_\Gamma \int_\mathcal{D} G\big(u(\mathbf{x},\vec{y})\big)\rho(\vec{y})\,d\mathbf{x}d\vec{y}$$

  – a spatial quadrature rule with the points $\mathbf{x}_r$ and weights W_r for $r = 1,\dots,R$ is used to approximate the spatial integral resulting in the approximation

$$\mathcal{J}(u) \approx \int_\Gamma \sum_{r=1}^{R} W_r G\big(u(\mathbf{x}_r,\vec{y})\big)\rho(\vec{y})\,d\vec{y}$$

  – a parameter-space quadrature rule with the points y_q and weights w_q for $q = 1,\dots,Q$ is used to approximate the spatial integral resulting in the approximation

$$\mathcal{J}(u) \approx \sum_{q=1}^{Q}\sum_{r=1}^{R} w_q W_r \rho(\vec{y}_q) G\big(u(\mathbf{x}_r,\vec{y}_q)\big)$$

– a set of points $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ is chosen in the parameter domain Γ

- these sample points are used to obtain the set of realizations $\{u_s(\mathbf{x})\}_{s=1}^{N_{sample}}$ of a finite element discretization of the SPDE

- each realization is determined by setting the parameters $\vec{y} = \vec{y}_s$ in the discretized SPDE

– if the probalistic quadrature points $\{\vec{y}\}_{q=1}^{Q}$ are the same as the sample points $\{\vec{y}\}_{s=1}^{N_{sample}}$, we directly define the computable approximation

$$\mathcal{J}(u) \approx \sum_{q=1}^{Q} \sum_{r=1}^{R} w_q W_r \rho(\vec{y}_q) G\big(u_q(\mathbf{x}_r)\big)$$

where we have, of course, renamed $u_s(\mathbf{x})$ by $u_q(\mathbf{x})$ since now they are one and the same

– if the the sample points $\{\vec{y}\}_{s=1}^{N_{sample}}$ are coarser than the probalistic quadrature points $\{\vec{y}\}_{q=1}^{Q}$, we first build a surrogate $G_{surrogate}(\mathbf{x}_r, \vec{y})$ for $G(\mathbf{x}_r, \vec{y})$

- the simplest means for doing this is to use the set of Lagrange interpolating polynomials $\{L_s(\vec{y})\}_{s=1}^{N_{sample}}$ corresponding to the sample points $\{\vec{y}_s\}_{s=1}^{N_{sample}}$, resulting in the surrogate approximation

$$G_{surrogate}(\mathbf{x}_r, \vec{y}) = \sum_{s=1}^{N_{sample}} G\big(u_s(\mathbf{x}_r)\big) L_s(\vec{y})$$

- other surrogate constructions may be used, e.g., least-squares fits to the data $\{\vec{y}_s, G\big(u_s(\mathbf{x}_r)\big)\}_{s=1}^{N_{sample}}$ using global orthogonal polynomials or even piecewise polynomials

- once the surrogate $G_{surrogate}(\mathbf{x}_r, \vec{y})$ has been constructed, one defines the indirect computable approximation

$$\mathcal{J}(u) \approx \sum_{q=1}^{Q} \sum_{r=1}^{R} w_q W_r \rho(\vec{y}_q) G_{surrogate}(\mathbf{x}_r, \vec{y}_q)$$

by evaluating the surrogate at the probabilistic quadrature points $\{\vec{y}_q\}_{q=1}^{Q}$

- for example, if the surrogate is constructed using Lagrange interpolating polynomials, we have the approximation

$$\mathcal{J}(u) \approx \sum_{s=1}^{N_{sample}} \sum_{r=1}^{R} W_r G\big(u_s(\mathbf{x}_r)\big) \sum_{q=1}^{Q} w_q \rho(\vec{y}_q) L_s(\vec{y}_q)$$

- of course, if the sample points $\{\vec{y}_s\}_{s=1}^{N_{sample}}$ are the same as the probabilistic quadrature points $\{\vec{y}_q\}_{q=1}^{N_q}$ so that $L_s(\vec{y}_q) = \delta_{sq}$, this approximation reduces to the one obtained before which, in this example, takes the simple form

$$\mathcal{J}(u) \approx \sum_{q=1}^{Q} w_q \rho(\vec{y}_q) \sum_{r=1}^{R} W_r G\big(u_q(\mathbf{x}_r)\big)$$

- Note that if one uses the sample points directly as quadrature points, then one does not need to construct a representation of the approximate solution in terms of the random parameters
  - if one uses a coarser set of sampling points relative to the quadrature points, one does have to build such a representation since it needs to be evaluated at the quadrature points, and not just the sample points
  - of course, this is also unlike the case for general SGMs in which one does build such a representation, e.g., an intrusive polynomial chaos expansion

- We will concentrate on the case where the sample points are used directly as quadrature points

- So, we next discuss quadrature rules that can be used to approximate quantities of interest
  - (coarser) versions of some of these rules can also supply sample points that can be used to build surrogates or response surfaces

- We will discuss quadrature rules for the N-dimensional hypercube, the case that most often arises in practice
  - other rectangular regions, i.e., bounding boxes, can be mapped in the obvious way to the unit hypercube

- Unfortunately, we do not have time to discuss sampling in unbounded domains or in general, non-rectangular domains

QUADRATURE RULES FOR HYPERCUBES

- One is tempted to use well-known quadrature rules to define the sample points for SSMs

- We will discuss two classes of quadrature rules for the N-dimensional hypercube
  - sampling and simple averaging rules
    - the canonical example is Monte Carlo integration
  - weighted quadrature rules based on standard one-dimensional rules
    - ultimately, we consider sparse grid Smolyak quadrature rules

- Recall that in the SSM framework we are using, the quadrature points are also the points used to sample the solutions of the SPDE

Sampling and simple averaging quadrature rules

- We consider sampling + simple averaging-based quadrature rules that are based on
  - determining a set of quadrature points $\{y_q\}_{q=1}^{Q}$
  - approximating integrals of a function $G(y)$ by an equal weight rule

$$\int_\Gamma G(\vec{y})\rho(y)\,dy \approx \frac{1}{Q}\sum_{q=1}^{Q} G(\vec{y}_q)$$

if one samples the points according to the PDF $\rho(\vec{y})$

or by

$$\int_\Gamma G(\vec{y})\rho(y)\,dy \approx \frac{1}{Q}\sum_{q=1}^{Q} \rho(y_q)G(y_q)$$

if one samples the points uniformly

- The second approach seems simpler, but is wasteful
  - the density of points is the same in regions where $\rho(\cdot)$ is small as where it is large
    - unfortunately, many sampling methods can only be used to sample uniformly or have difficulty, i.e., they are much less efficient, when sampling nonuniformly

- Note that the weights do not depend on the position of the points $\{\vec{y}_q\}_{q=1}^{Q}$ or on other geometric quantities

Monte Carlo sampling

- As has already been said, the simplest quadrature rule is based on Monte Carlo, i.e., random, sampling of the hypercube
  - random sampling could be done uniformly in the hypercube
    - in which case $w_q = \dfrac{\rho(\vec{y}_q)}{Q}$
  - random sampling could instead be done according to the density function $\rho(\vec{y})$ by, e.g., a rejection method
    - in which case $w_q = \dfrac{1}{Q}$
- Monte Carlo integration has one very great virtue (other than its simplicity)
  - its convergence behavior is independent of the dimension N, i.e., of the number of parameters

- Unfortunately, it also has one great fault
  - its convergence behavior is slow $\quad \text{Error} = O\Big(\frac{\sigma}{\sqrt{Q}}\Big)$
- The slow convergence of Monte Carlo integration has motivated the huge amount of effort devoted to improving or replacing Monte Carlo sampling as an integration rule
  - it has also motivated the development of stochastic Galerkin methods

“Improved” sampling + simple averaging-based quadrature rules

- There have been many sampling + simple averaging-based quadrature rules proposed as replacements for Monte Carlo quadrature, including

  variance reduction Monte Carlo methods

  quasi-Monte Carlo methods (Halton, Sobol, Faure, Hammersley, . . .)

  stratified sampling

  Latin hypercube sampling and its many “improved” versions

  orthogonal arrays

  lattice rules

  importance sampling

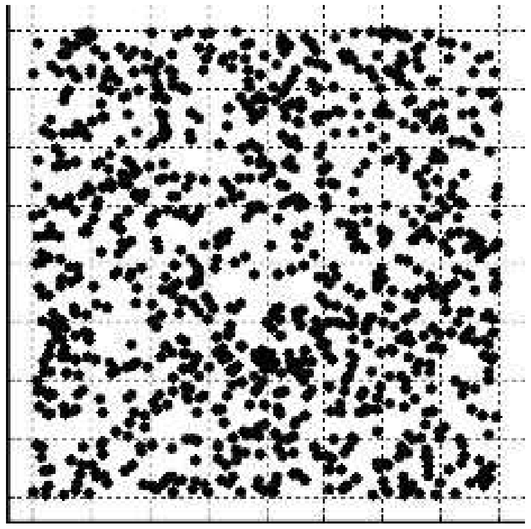
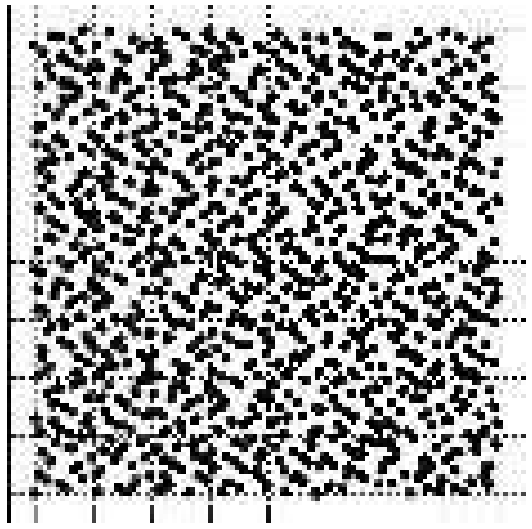
  etc.

- In general, these “improved” rules have, in theory, improved rates of convergence, at least for not too large N
  - the best theoretical result is of the type

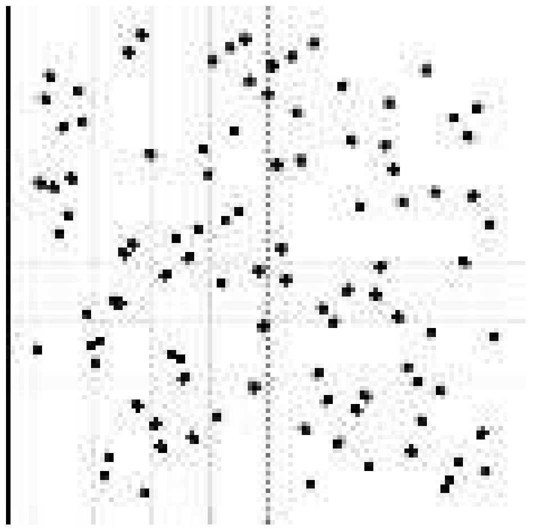
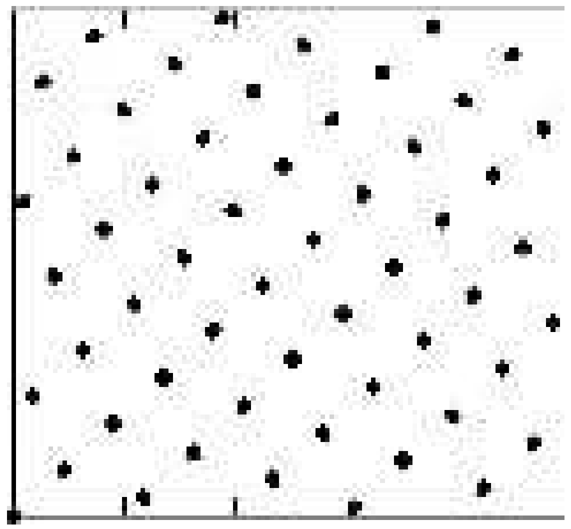
$$\text{Error} = O\Big(\frac{(\ln Q)^N}{Q}\Big) \qquad \Longleftarrow \text{note the dependence on } N$$

  - this is often a pessimistic estimate
  - for large N, the $(\ln Q)^N$ term dominates
    - the curse of dimensionality is still with us
  - also, in many cases, biasing problems exist, especially for a large number of sample points

- However, if one is careful when using them, the “improved” sampling and averaging methods often can indeed improve on Monte-Carlo sampling

Monte Carlo and quasi-Monte Carlo point sets

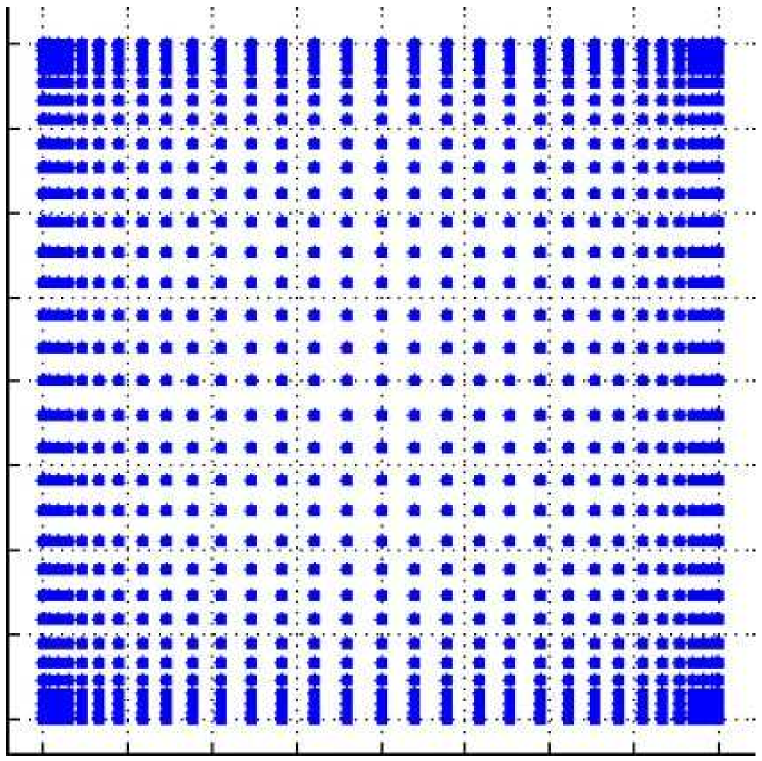
Latin hypercube and lattice rule point sets

Tensor products of standard 1-D quadrature rules

- One is familiar with many quadrature rules in 1D
- On the hypercube, one can easily define multiple integration rules as tensor products of 1D rules
- As we have already seen, tensor products really suffer from the curse of dimensionality
- Tensor product rules integrate tensor products of polynomials exactly
- Just as was the case for interpolation and approximation, one can get the same rate of convergence using quadrature rules that integrate complete polynomials exactly
- The same table of numbers used before applies here

Quadrature rules in hypercubes

$N =$ no. random parameters	number of quadrature points in each direction	$Q =$ number of quadrature points using complete polynomial rule	$Q =$ number of quadrature points using a tensor product rule
3	4	20	64
	6	56	216
5	4	56	1,024
	6	252	7,776
10	4	286	1,048,576
	6	3,003	60,046,176
20	4	1,771	$> 1\times10^{12}$
	6	53,130	$> 3\times10^{15}$
100	4	176,851	$> 1\times10^{60}$
	6	96,560,646	$> 6\times10^{77}$



A tensor product set of quadrature points in 2D

- On the other hand, tensor product rules are easy to define
  - the quadrature points are tensor products of the quadrature points of the 1D rules
  - the quadrature weights are products of the weights of the 1D rules

- High-dimensional rules based on complete polynomials are not so easy to define
  - determining a good set of quadrature points and the corresponding quadrature weights is difficult
  - these difficulties further motivated interest in SGM methods

- But now, there is available an intermediate means of defining quadrature rules
  - the number of points is much less that that for tensor product rules, but is somewhat greater than that for complete polynomial rules
  - these rules are constructed through judicious sparsifications of tensor product rules
  - the are known as Smolyak or sparse grid quadrature rules

SPARSE (SMOLYAK) QUADRATURE RULE-BASED STOCHASTIC SAMPLING METHODS

- Let I be a positive integer and for each $i = 1, \dots, I$, let m_i denote a positive integer

- For each $i = 1, \dots, I$, let $\Theta^{(i)} = \{y_1^{(i)}, \dots, y_{m_i}^{(i)}\}$ denote a set of points in $[-1, 1]$
  - note that for convenience, we will be looking at the hypercube $[-1, 1]^N$

- Let $N > 1$ denote the number of parameters

- Let $p = (p_1, p_2, \dots, p_N)$ denote a multi-index,
  - in this case, an N-vector whose components are positive integers

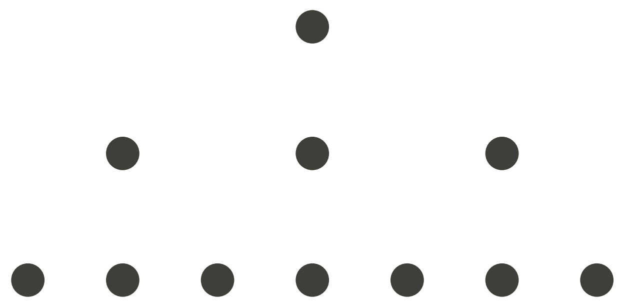
  and let $|p| = \sum_{n=1}^{N} p_n$

- Let M denote a positive integer
- Let $\mathcal{I}(M,N) = \{p \ : \ M+1 \le |p| \le N+M\}$
- Then,

$$\mathcal{S}(M,N) = \bigcup_{p \in \mathcal{I}(M,N)} \Theta^{(p_1)} \otimes \Theta^{(p_2)} \otimes \cdots \otimes \Theta^{(p_N)}$$

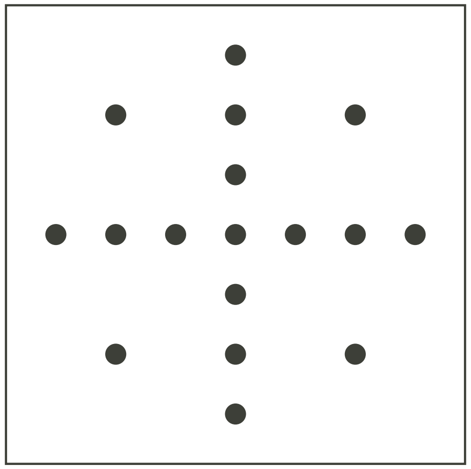
defines a sparse grid

- Example
  - let $I = 3$, $m_1 = 1$, $m_2 = 3$, and $m_3 = 7$
  - let $\Theta^{(i)}$, $i = \ldots, I = 3$ be given by the three one-dimensional nested point sets

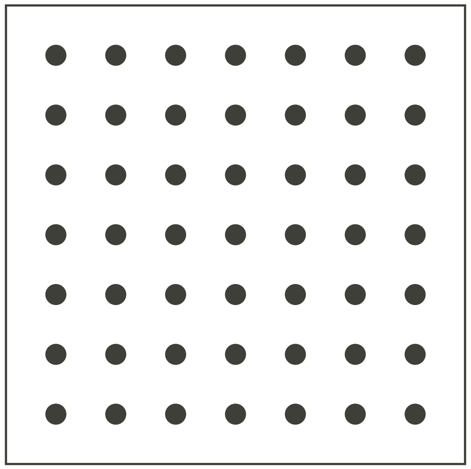


  - let $N = 2$ and $M = 2$ so that $\mathcal{I}(2,2) = \{p \ : \ 3 \leq |p| \leq 4\}$
  - $\mathcal{I}(2,2)$ then contains the combinations
$$(p_1, p_2) = (1,1),\ (1,2),\ (2,1),\ (3,1),\ (1,3),\ (2,2)$$
but not the combinations
$$(p_1, p_2) = (2,3),\ (3,2),\ (3,3)$$
    - for nested point sets, it is enough to include the combinations for which $|p| = N + M$, i.e., $(3,1),\ (1,3),\ (2,2)$ in the example

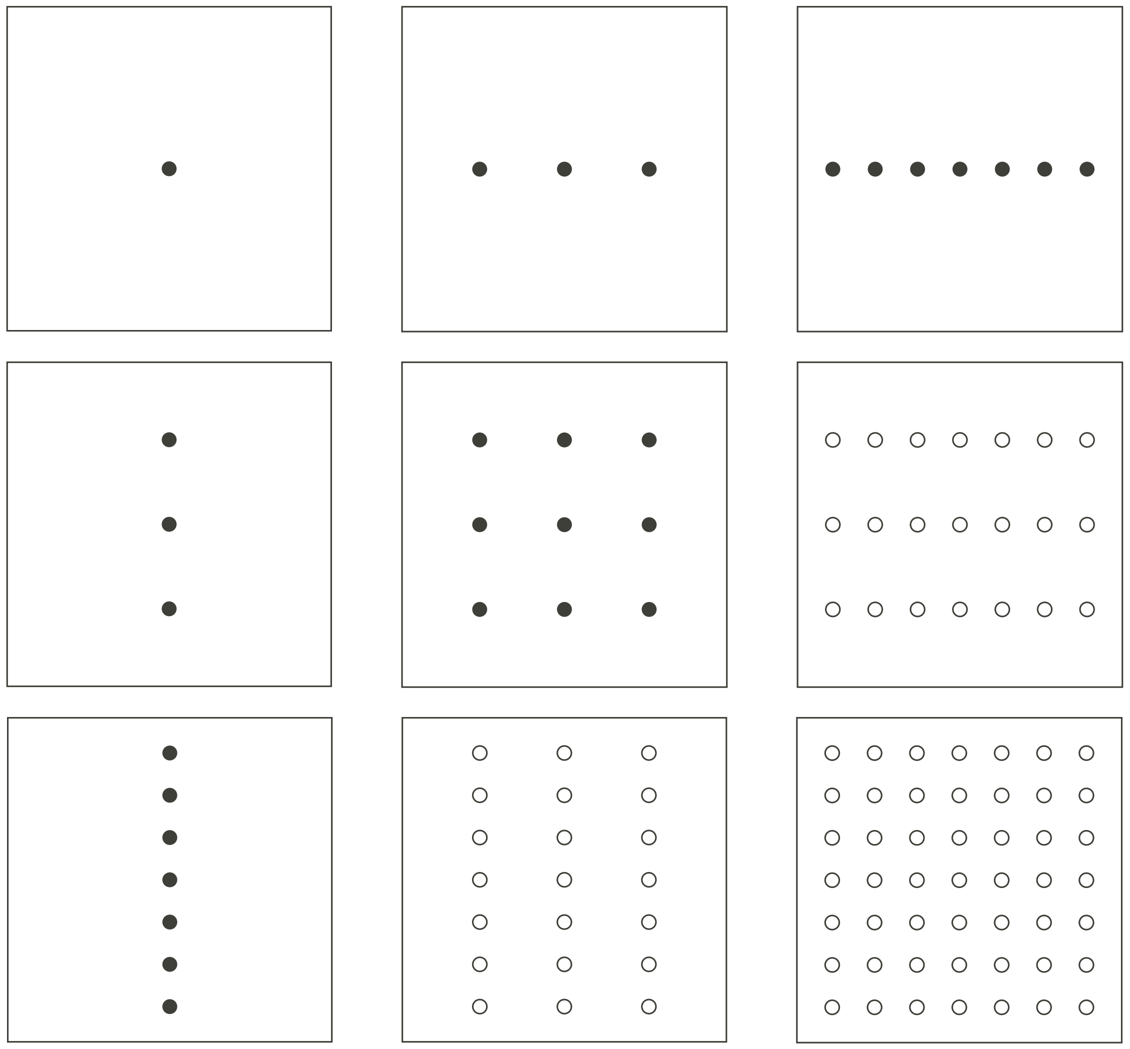
– then, $\mathcal{S}(2,2)$ is given by



– this should be contrasted with the full tensor-product point set



- the following diagram shows how the sparse grid comes about



● point sets included in $\mathcal{S}(2,2)$ ○ point sets not included in $\mathcal{S}(2,2)$

- What Smolyak showed is that

  – if one chooses the underlying one-dimensional grids to be the quadrature points for some integration rule

  then

  – the accuracy of the full tensor product point set can be preserved with point sets with much fewer points

- Along the way, Smolyak also showed how to systematically compute the weights of the resulting sparse quadrature rule

- The use of Smolyak grids in the SPDE setting has been rigorously analyzed for some simple linear and nonlinear elliptic PDEs

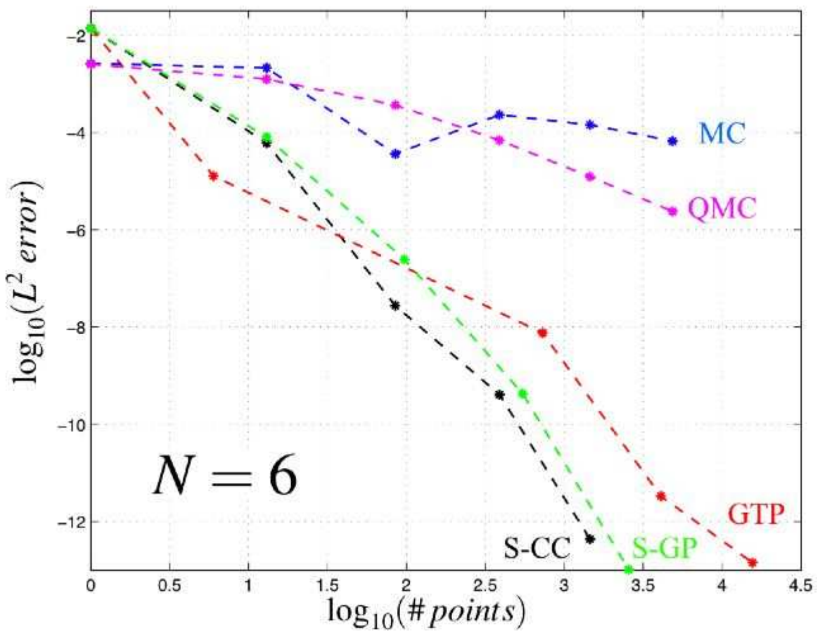
- Some choices of one-dimensional quadrature rules upon which the Smolyak grids can be constructed
  - Newton-Cotes: nested equidistant abscissas by taking $m_1 = 1$ and $m_i = 2^{i-1} + 1$ for $i > 1$
    - maximum degree of exactness is $m_I - 1$
    - can have (highly) negative weights causing numerical inaccuracies
  - Clenshaw-Curtis: nested (same growth as above) Chebyshev points
    - maximum degree of exactness is $m_I - 1$
    - nested grids keep the number of points down
  - Gauss: non-nested abscissas
    - maximum degree of exactness is $2m_I - 1$
  - Gauss-Patterson: seems to have good promise

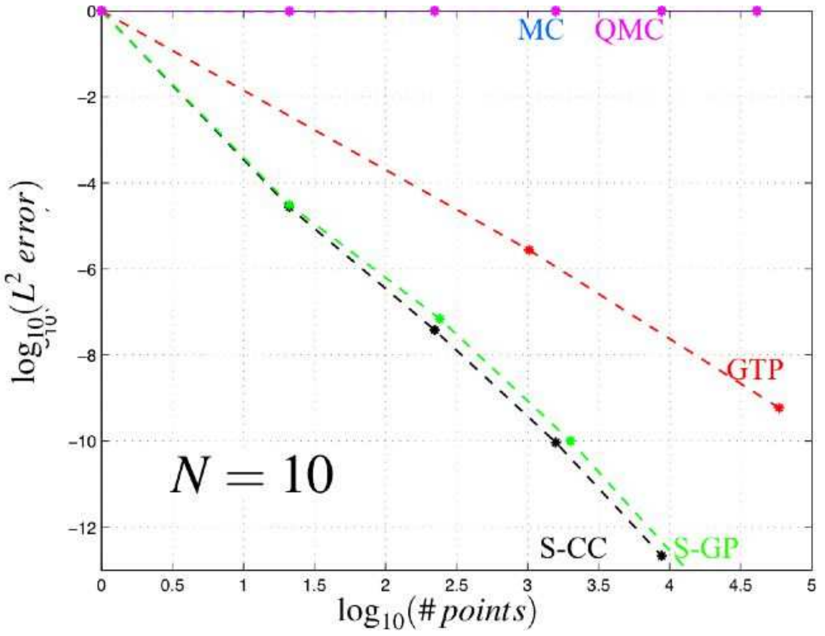
Results that follow are from papers of Nobile, Tempone, and Webster

- For the integral

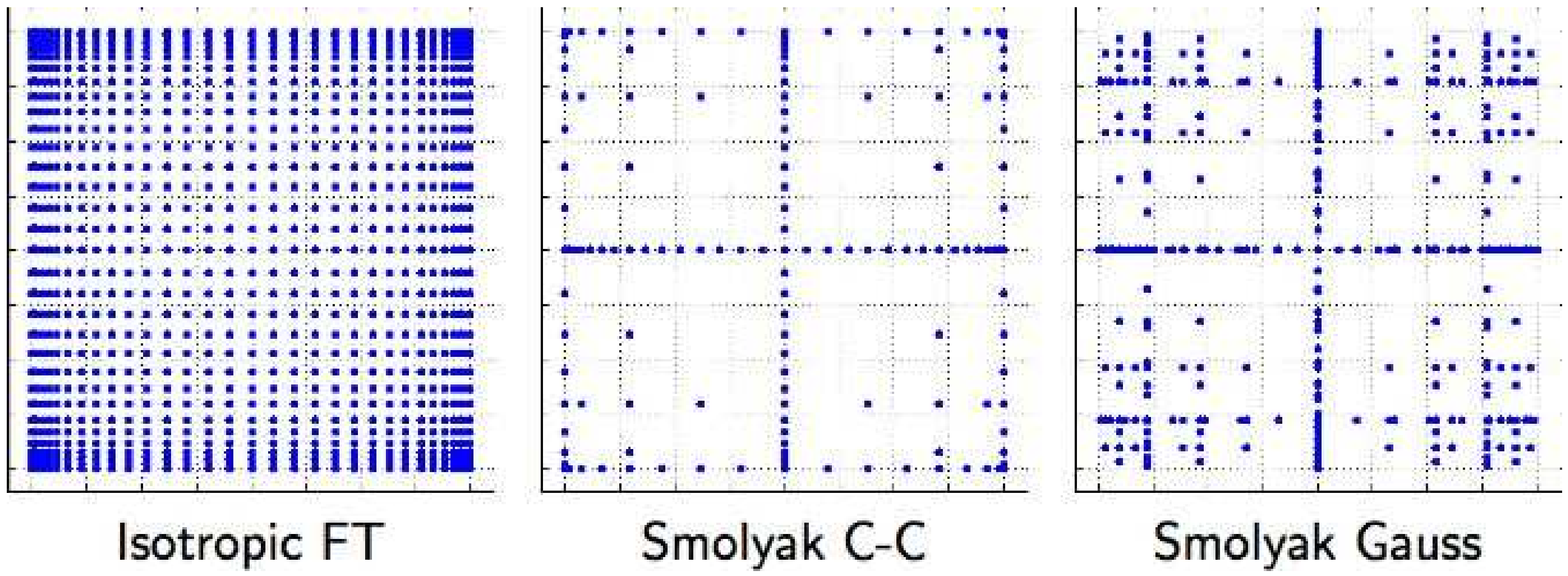
$$\int_{\mathbb{R}^N} \exp\left(-\sum_{n=1}^{N} a_n^2 (y_n - b_n)^2\right) d\vec{y}$$

where a_n and b_n are randomly sampled uniformly in $(0,1)$, we have the following errors for different quadrature rules

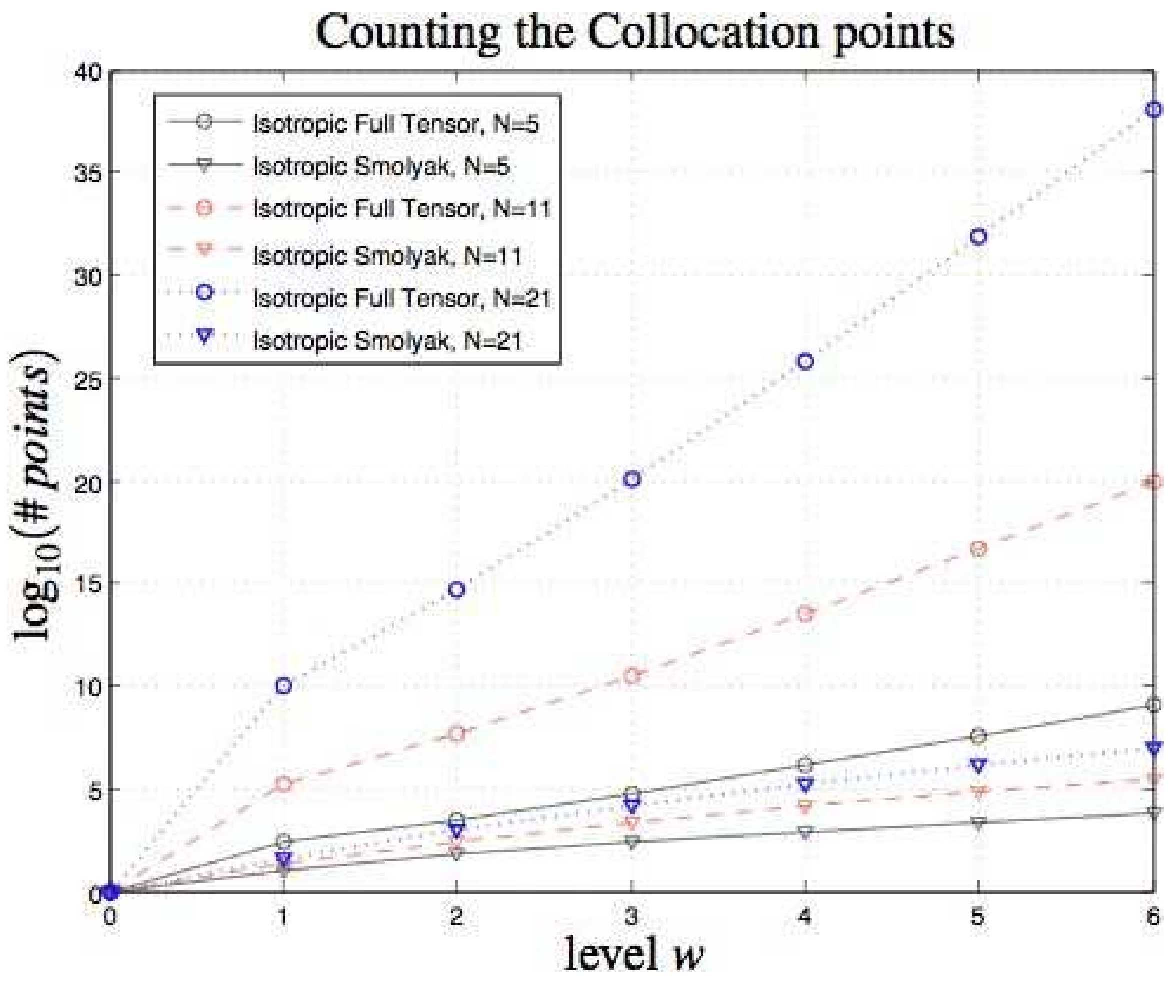




Comparisons of errors vs. number of quadrature points for different integration rules



For $N = 2$ and $M = 5$: comparison of full tensor product grids with two Smolyak grids



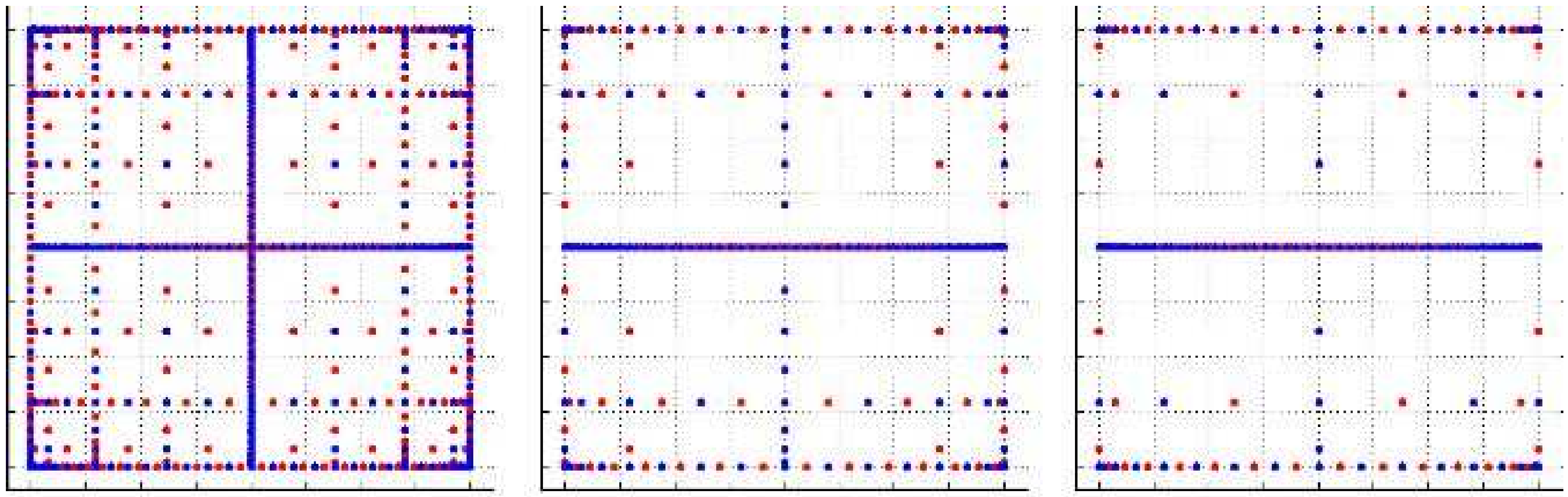
For $N = 5$, 11, and 21: comparison of full tensor product grids with Clenshaw-Curtis-Smolyak grids for different levels, i.e., for different maximum number of points in each direction

- There is more good news about Smolyak grids

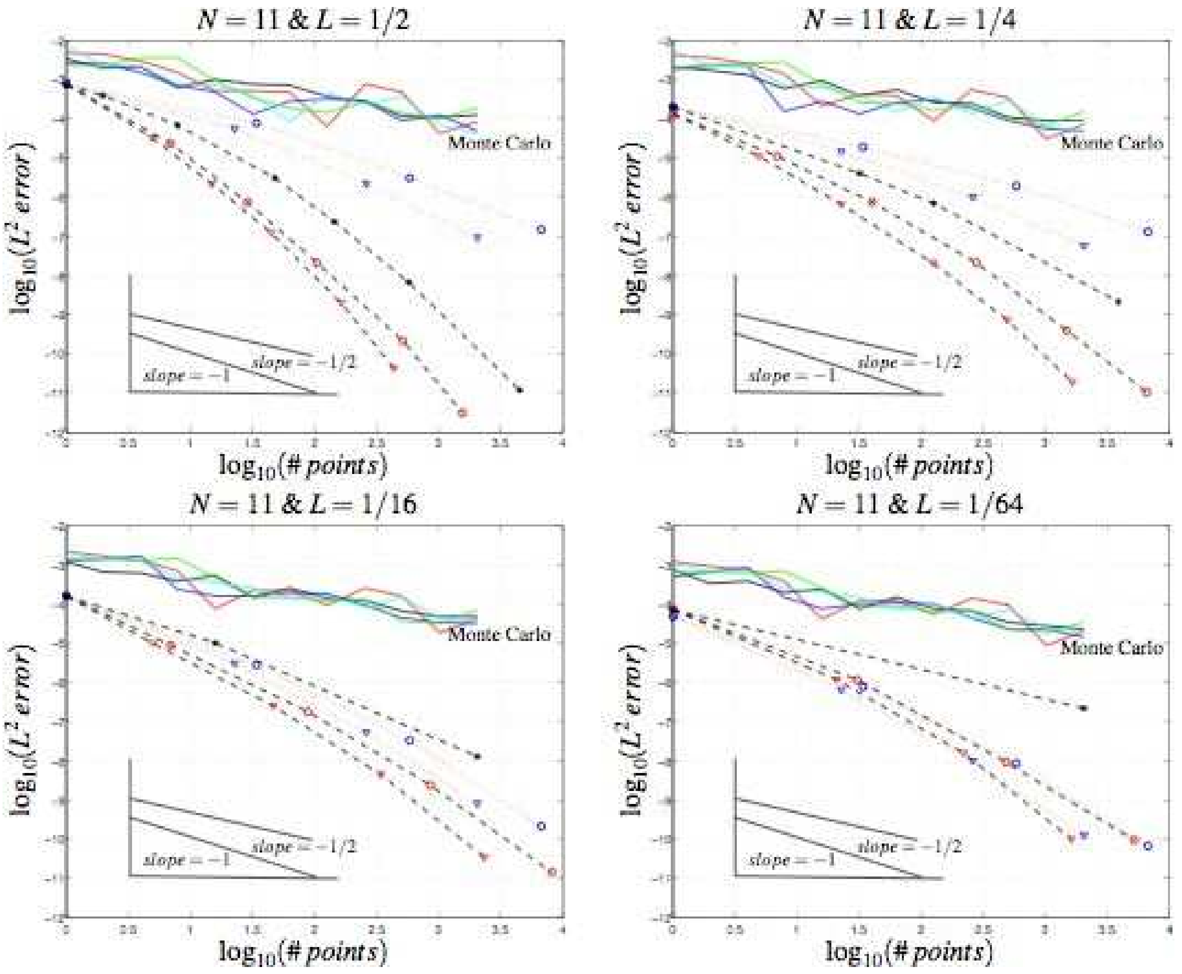
- Recently, anisotropic Smolyak grids have been developed to take advantage of anisotropies in the relative importance of random parameters

- For example, in the Karhunen-Loéve expansion for the colored noise case, the random variables $y_1, y_2, \ldots$ are increasingly less influential

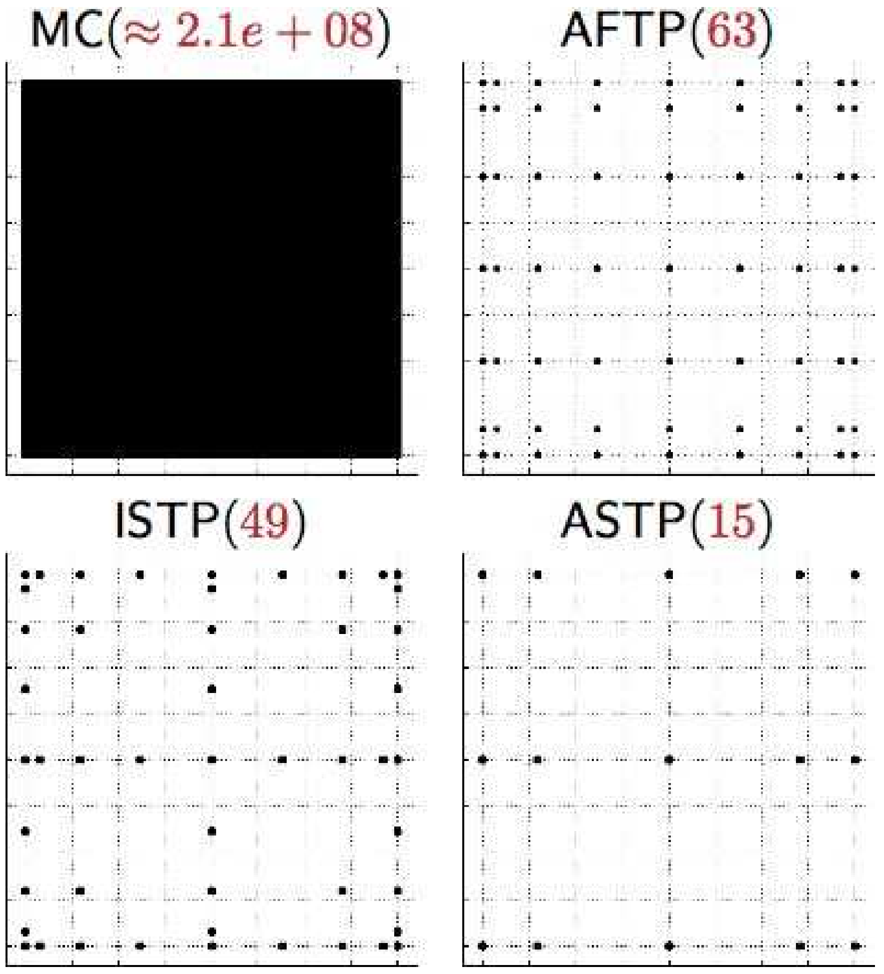
- Adaptive strategies have been developed to determine how to take advantage of such anisotropies


$\alpha_2/\alpha_1 = 1$ $\alpha_2/\alpha_1 = 1.5$ $\alpha_2/\alpha_1 = 2$

Anisotropic Clenshaw-Curtis sparse grids for different levels of anisotropy; on the left is the isotropic case; the anisotropic grids will yield the same accuracy as the isotropic one, provided the integrand possesess the necessary anisotropy



L^2 errors in the expected values of the solution of an SPDE using different sampling strategies; Monte Carlo is always worst, anisotropic Smolyak best, with Clenshaw-Curtis being better than Gauss; L is a correlation length for the colored noise



Number of points needed to reduce to reduce the L^2 errors in the expected values of the solution of an SPDE by a factor of 10^4

- This shows the effectiveness of using stochastic sampling methods along with modern sparse grid techniques

LOCAL POLYNOMIAL APPROXIMATING SPACES IN STOCHASTIC GALERKIN METHODS

PIECEWISE POLYNOMIAL APPROXIMATING SPACES FOR PARAMETER SPACE DISCRETIZATION

- Emulating finite element spatial discretization methods, one is led to locally-supported piecewise polynomial spaces for approximating functions of the random parameters

- One starts by "triangulating" Γ, the set of all possible values for the random parameters $\{y_1, \ldots, y_N\}$
  - of course, unless one wants to get fancy, i.e.,
    - use infinite elements or other methods for treating unbounded domains

    we have to assume that Γ is bounded
  - thus, we consider problems for which the Γ_n, $n = 1, \ldots, N$, themselves are bounded
    - e.g., we cannot consider y_1 to be a Gaussian random parameter since, in this case, $\Gamma_1 = (-\infty, \infty)$
    - of course, we can considered truncated Gaussian parameters

- One then chooses Z_K to be a space of piecewise polynomial functions of degree less than of equal to M, defined with respect to the triangulation
  - since $Z_K \subset L^q_\rho(\Gamma)$, one can choose $M = 0$, i.e., piecewise constant functions
  - however, one can choose higher degree piecewise polynomials as well
  - one is free to choose discontinuous finite element spaces

- Unfortunately, the number of parameters N cannot be large
  - even for a subdivision with two elements in each direction, N cannot be big, e.g., $K = 2^N$ becomes prohibitively large very quickly

- Also, triangulating in high dimensions is not an easy task
  - unless N is small, one can in practice only consider the case of Γ being rectangular domain in $\mathbb{R}^N$ that is "triangulated" into smaller rectangular domains

- One can choose a standard "finite element"-type basis set
  - $\{\psi_k(\vec{y})\}_{k=1}^K$ consists of compactly supported piecewise polynomials
  - if Z_K is a discontinuous (with respect to the triangulation of Γ) finite element space, then each basis function can be chosen to have support over only a single element
  - if Z_K is a continuous (with respect to the triangulation of Γ) finite element space, then each basis function can be chosen to have support over a small patch of elements

- There is a really big difference between using discontinuous and continuous finite element-type spaces to discretize in parameter space

- First, consider an example of a continuous finite element-type space
  - Γ is a hypercube in N-dimensions ($N =$ number of random parameters)
  - Γ is subdivided into $N_{hypercubes}$ smaller hypercubes
  - Z_K consists of tensor products of continuous piecewise polynomials of degree less that or equal to $M \geq 1$ in each parameter direction
  - then, the number of probabilistic degrees of freedom is given by
$$K = \left(M N_{hypercubes}^{1/N} + 1 \right)^N$$
  - as always, the discrete problem involves JK degrees of freedom $c_{j,k}$

- If we look at the $JK \times JK$ coefficient matrix for the discrete system (emanating from a linear Poisson problem)

$$\int_\Gamma \int_\mathcal{D} a(\mathbf{x};\vec{y})\nabla\phi_j(\mathbf{x}) \cdot \nabla\phi_{j'}(\mathbf{x})\, \psi_k(\vec{y})\psi_{k'}(\vec{y})\rho(\vec{y})\, d\mathbf{x}d\vec{y}$$

  we see that it is sparse with respect to both the spatial and probabilistic indices

  – if the support of $\phi_j(\mathbf{x})$ and $\phi_{j'}(\mathbf{x})$ do not overlap, then the corresponding matrix entry vanishes for all k and k'

  – if the support of $\psi_k(\mathbf{x})$ and $\psi_{k'}(\mathbf{x})$ do not overlap, then the corresponding matrix entry vanishes for all j and j'

  – this sparsity can be taken advantage of when one solves the system, especially if one uses an iterative method

  – however, we still have a coupled (albeit sparse) $JK \times JK$ system to solve

- Now, consider an example of using discontinuous finite element-type spaces to discretize in parameter space
  - Γ is a hypercube in N-dimensions ($N =$ number of random parameters)
  - Γ is subdivided into $N_{hypercubes}$ smaller hypercubes
  - in each element, Z_K consists of complete polynomials of degree less that or equal to $M \geq 0$
    - no continuity is required across element boundaries
  - then, the number of probabilistic degrees of freedom is given by

$$K = N_{hypercubes} \left(\frac{(N+M)!}{N!M!} \right)$$

    which can be larger than that obtained using continuous finite element-type spaces
  - as always, the discrete problem involves JK degrees of freedom $c_{j,k}$

Piecewise polynomial approximation in parameter space

$N =$ no. random parameters	$M =$ maximal degree of polynomials	$N_{hypercubes}^{1/N} =$ no. of intervals in each direction	$K =$ no. of probabilistic degrees of freedom: continuous tensor product basis	$K =$ no. of probabilistic degrees of freedom: discontinuous basis
3	0	5	–	125
		10	–	1,000
	1	5	216	500
		10	1,331	4,000
	2	5	1,331	1,250
		10	9,261	10,000
5	0	5	–	3,125
		10	–	100,000
	1	5	7,776	18,750
		10	161,051	600,000
	2	5	161,051	65,625
		10	4,084,101	2,100,000

- But, let's examine the $JK \times JK$ coefficient matrix for the discrete system in the discontinuous finite element case

$$\int_\Gamma \int_{\mathcal{D}} a(\mathbf{x};\vec{y}) \nabla\phi_j(\mathbf{x}) \cdot \nabla\phi_{j'}(\mathbf{x})\, \psi_k(\vec{y})\psi_{k'}(\vec{y})\rho(\vec{y})\, d\mathbf{x}d\vec{y}$$

  – again, we have the usual sparsity with respect to both the spatial indices

  – but now, since the support of the probabilistic basis functions $\{\psi_k(\vec{y})\}_{k=1}^K$ is restricted to a single element in parameter space, we have that

    - with respect to parameter space, the matrix is block diagonal

    - there is a complete uncoupling of the probabilistic degrees of freedom

- Let $\Gamma_{hypercube}$ denote one of the $N_{hypercubes}$ elements in the subdivision of Γ into smaller hypercubes

- Let $K_{hypercube}$ denote the probabilistic degrees of freedom in each element $\Gamma_{hypercube}$, i.e.,

$$K_{hypercube} = \frac{(N+M)!}{N!M!} = \frac{K}{N_{hypercubes}}$$

- For each of the $N_{hypercubes}$ elements $\Gamma_{hypercube}$, let

$$I_{hypercube} = \Big\{ k \in \{1, \ldots, K\} \mid \text{supp}\big(\psi_k(\vec{y})\big) \subset \Gamma_{hypercube} \Big\}$$

  – note that the cardinality of the index set $I_{hypercube}$ is $K_{hypercube}$

- Then, the coupled $JK \times JK$ system for the degrees of freedom $c_{j,k}$ uncouples into $N_{hypercubes}$ systems, each of size $JK_{hypercube} \times JK_{hypercube}$

$$\int_{\mathcal{D}}\int_{\Gamma} \rho(\vec{y}) S\Big(\sum_{j=1}^{J}\sum_{k=1}^{K} c_{jk}\phi_j(\mathbf{x})\psi_k(\vec{y}), \vec{y}\Big) T\Big(\phi_{j'}(\mathbf{x})\Big)\psi_{k'}(\vec{y})\, d\mathbf{x}d\vec{y}$$
$$= \int_{\mathcal{D}}\int_{\Gamma} \rho(\vec{y})\phi_{j'}(\mathbf{x})\psi_{k'}(\vec{y}) f(\vec{y})\, d\mathbf{x}d\vec{y}$$

for $j' \in \{1,\dots,J\}$ and $k' \in \{1,\dots,K\}$

$$\sum_{j=1}^{J}\sum_{k\in I_{hypercube}} c_{j,k} \int_{\Gamma_{hypercube}}\int_{\mathcal{D}} a\nabla\phi_j(\mathbf{x})\cdot\nabla\phi_{j'}(\mathbf{x})\,\psi_k(\vec{y})\psi_{k'}(\vec{y})\rho(\vec{y})\, d\mathbf{x}d\vec{y}$$
$$= \int_{\Gamma_{hypercube}}\int_{\mathcal{D}} f\phi_{j'}(\mathbf{x})\psi_{k'}(\vec{y})\rho(\vec{y})\, d\mathbf{x}d\vec{y}$$

for all $j' = 1,\dots,J$ and $k' \in I_{hypercube}$

- The moral of the story is that, in practice, once pretty much has to settle for piecewise constant approximations in parameter space

- Even for this case, N cannot be too large

PIECEWISE CONSTANT APPROXIMATING SPACES

- Let $\cup_{k=1}^{K}\Gamma_k$ denote a subdivision of Γ into disjoint, non-overlapping subsets
  - we have that

$$\cup_{k=1}^{K}\overline{\Gamma}_k = \overline{\Gamma} \qquad \text{and} \qquad \Gamma_k \cap \Gamma_{k'} = \emptyset \quad \text{if } k \neq k'$$

- Let

$$\psi_k(\vec{y}) = \begin{cases} 1 & \text{if } \vec{y} \in \Gamma_k \\ 0 & \text{otherwise} \end{cases} \qquad \text{for } k \in \{k, \ldots, K\}$$

  and let

$$Z_K = \text{span}\,\{\psi_k\}_{k=1}^{K}$$

  - thus, Z_K is the space of piecewise constant functions with respect to the partition $\cup_{k=1}^{K}\Gamma_k$ of Γ

- Clearly, $Z_K \subset L^p_\rho(\Gamma)$ so that it can be used as an approximating space for discretizing parameter dependences of solution of an SPDE

- Recall that, after the invocation of the piecewise constant basis functions and of a parameter-space quadrature rule, the stochastic Galerkin method has the form

$$\sum_{r=1}^{R} w_r \rho(\vec{y}_r) \psi_{k'}(\vec{y}_r) \int_{\mathcal{D}} S\Big(\sum_{j=1}^{J} \sum_{k=1}^{K} c_{jk} \phi_j(\mathbf{x}) \psi_k(\vec{y}_r), \vec{y}_r \Big) T\Big(\phi_{j'}(\mathbf{x}) \Big)\, d\mathbf{x}$$
$$= \sum_{r=1}^{R} w_r \rho(\vec{y}_r) \psi_{k'}(\vec{y}_r) \int_{\mathcal{D}} \phi_{j'}(\mathbf{x}) f(\vec{y}_r)\, d\mathbf{x}$$
$$\text{for } j' \in \{1, \ldots, J\} \text{ and } k' \in \{1, \ldots, K\}$$

where $\{w_r, \vec{y}_r\}_{r=1}^{R}$ denotes the quadrature rule used to approximate integrals over parameter space Γ

- Suppose we choose the quadrature rule so that

$$R = K \qquad \text{and} \qquad \vec{y}_r \in \Gamma_r \quad \text{for } r \in \{1, \ldots, R = K\}$$

  – thus,

    - each quadrature point $\vec{y}_r$ belongs to one of the subsets Γ_k

    and

    - each subset contains one and only one of the quadrature points

  – Clearly, we then have that

$$\psi_k(\vec{y}_r) = \delta_{kr} \qquad \text{for all } k, r \in \{1, \ldots, K = R\}$$

- Then, the discretized stochastic Galerkin system reduces to

$$\int_{\mathcal{D}} S\Big(u_r(\mathbf{x}), \vec{y}_r\Big) T\Big(\phi_{j'}(\mathbf{x})\Big)\, d\mathbf{x} = \int_{\mathcal{D}} \phi_{j'}(\mathbf{x}) f(\vec{y}_r)\, d\mathbf{x}$$

for $j' \in \{1, \ldots, J\}$ and $r \in \{1, \ldots, R = K\}$

where $u_r(\mathbf{x}) = \sum_{j=1}^{J} c_{jr}\phi_j(\mathbf{x})$

– thus, we have total uncoupling of the spatial and parameter problems

– we solve a sequence of $R = K$ problems of size J to determine $\{u_r(\mathbf{x})\}_{r=1}^{R}$

– then, the stochastic Galerkin-piecewise constant approximation of the solution of the SPDE is simply given by

$$u(\mathbf{x}; \vec{y}) = u_r(\mathbf{x}) \qquad \text{for } \vec{y} \in \Gamma_r$$

- Note that to determine the $u_r(\mathbf{x})$ one does not have to explicitly know the weights w_r or the subregions Γ_k
  - one need only know the point set $\{\vec{y}_k\}_{k=1}^K$

- Note also that there is no restrictions on the point set $\{\vec{y}_k\}_{k=1}^K$
  - one can, in fact, use any of the point sets we have encountered in discussing stochastic sampling or stochastic collocation or stochastic Galerkin methods

- Clearly,

  any stochastic sampling method can be viewed as a stochastic Galerkin method

Approximations of quantities of interest

- It is natural to use the same quadrature rule

  - to approximate a quantity of interest

  as was used to

  - approximate the integrals in discretized SPDE,

  i.e., we choose

$$K = R = Q$$

$$\{\vec{y}_k\}_{k=1}^{K} = \{\vec{y}_r\}_{r=1}^{R} = \{\vec{y}_q\}_{q=1}^{Q} \qquad \text{and} \qquad \{w_r\}_{r=1}^{R} = \{w_q\}_{q=1}^{Q}$$

- We then have that

$$\psi_r(\vec{y}_q) = \delta_{rq} \qquad \text{for all } r, q \in \{1, \ldots, K = R = Q\}$$

- Using this in the expression for the approximation of a quantity of interest results in

$$\int_\Gamma G\Big(u(\mathbf{x};\vec{y})\Big)\rho(\vec{y})\,d\vec{y} \approx \sum_{q=1}^{Q} w_q\rho(\vec{y}_q)G\Big(u_{SC}(\mathbf{x})\Big)$$
$$= \sum_{q=1}^{Q} w_q\rho(\vec{y}_q)G\Big(\sum_{r=1}^{R} u_r(\mathbf{x})\psi_r(\vec{y}_q)\Big) = \sum_{q=1}^{Q} w_q\rho(\vec{y}_q)G\Big(u_q(\mathbf{x})\Big)$$

i.e.,

$$\int_\Gamma G\Big(u(\mathbf{x};\vec{y})\Big)\rho(\vec{y})\,d\vec{y} \approx \sum_{q=1}^{Q} w_q\rho(\vec{y}_q)G\Big(u_q(\mathbf{x})\Big)$$

where, for $q \in \{1,\ldots,Q = R = K_{LI}\}$, $u_q(\mathbf{x}) = \sum_{j=1}^{J} c_{jq}\phi_j(\mathbf{x})$ is determined from

$$\int_{\mathcal{D}} S\Big(u_q(\mathbf{x}),\vec{y}_q\Big)T\Big(\phi_{j'}(\mathbf{x})\Big)\,d\mathbf{x} = \int_{\mathcal{D}} \phi_{j'}(\mathbf{x})f(\vec{y}_q)\,d\mathbf{x} \qquad \text{for } j' \in \{1,\ldots,J\}$$

- This all looks very familiar:
  - it looks just the same as when we discussed stochastic collocation methods
  - in fact, there is very little distinction between stochastic sampling and stochastic collocation methods
  - and, as we have seen, all stochastic sampling and stochastic collocation methods can be derived from the stochastic Galerkin framework

ECONOMIES IN POLYNOMIAL CHAOS METHODS FOR LINEAR SPDES

- Suppose that the SPDE is linear in the solution u

- For example, consider the case for which one has, after using a polynomial chaos expansion method, the SPDE[†]

$$\int_{\mathcal{D}}\int_{\Gamma}\rho(\vec{y})a(\mathbf{x};\vec{y})S\Big(\sum_{j=1}^{J}\sum_{k=1}^{K_{PC}}c_{jk}\phi_j(\mathbf{x})\Psi_k(\vec{y})\Big)T\Big(\phi_{j'}(\mathbf{x})\Big)\Psi_{k'}(\vec{y})\,d\mathbf{x}d\vec{y}$$
$$=\int_{\mathcal{D}}\int_{\Gamma}\rho(\vec{y})\phi_{j'}(\mathbf{x})\Psi_{k'}(\vec{y})f(\mathbf{x};\vec{y})\,d\mathbf{x}d\vec{y},$$

where now both $S(\cdot)$ and $T(\cdot)$ are linear

[†]Here, it is useful to follow the explicit dependences of the data functions a and f on the spatial variable $\mathbf{x}$

- Since, $S(\cdot)$ is linear and does not involve derivatives with respect to the components of $\vec{y}$, we have that

$$\sum_{j=1}^{J}\sum_{k=1}^{K_{PC}} c_{jk}\int_{\mathcal{D}} S\Big(\phi_j(\mathbf{x})\Big)T\Big(\phi_{j'}(\mathbf{x})\Big)\int_{\Gamma} a(\mathbf{x};\vec{y})\rho(\vec{y})\Psi_k(\vec{y})\Psi_{k'}(\vec{y})\,d\vec{y}d\mathbf{x}$$
$$= \int_{\mathcal{D}} \phi_{j'}(\mathbf{x})\int_{\Gamma} f(\mathbf{x};\vec{y})\rho(\vec{y})\Psi_{k'}(\vec{y})\,d\vec{y}d\mathbf{x}$$

- In this linear SPDE case, there are two economies possible in the implementation of PC methods

PC-expansions of data functions

- We approximate the data functions a and f in the same way one approximates the solution, i.e., using PC-expansions

  – thus, we assume we have in hand the approximations

$$a(\mathbf{x};\vec{y}) \approx \sum_{k''=1}^{K_{PC}} a_{k''}(\mathbf{x})\Psi_{k''}(\vec{y})$$

and

$$f(\mathbf{x};\vec{y}) \approx \sum_{k''=1}^{K_{PC}} f_{k''}(\mathbf{x})\Psi_{k''}(\vec{y})$$

– substituting into the PC-discretization of the SPDE results in

$$\sum_{k''=1}^{K_{PC}} \sum_{j=1}^{J} \sum_{k=1}^{K_{PC}} c_{jk} \left(\int_{\mathcal{D}} a_{k''}(\mathbf{x}) S\Big(\phi_j(\mathbf{x})\Big) T\Big(\phi_{j'}(\mathbf{x})\Big)\, d\mathbf{x} \right) \left(\int_{\Gamma} \rho(\vec{y}) \Psi_k(\vec{y}) \Psi_{k'}(\vec{y}) \Psi_{k''}(\vec{y})\, d\vec{y} \right)$$

$$= \sum_{k''=1}^{K_{PC}} \left(\int_{\mathcal{D}} f_{k''}(\mathbf{x}) \phi_{j'}(\mathbf{x})\, d\mathbf{x} \right) \left(\int_{\Gamma} \rho(\vec{y}) \Psi_{k'}(\vec{y}) \Psi_{k''}(\vec{y})\, d\vec{y} \right)$$

$$= \int_{\mathcal{D}} f_k(\mathbf{x}) \phi_{j'}(\mathbf{x})\, d\mathbf{x}$$

where the last equality follows from the orthonormality of the PC-basis functions $\{\Psi_k(\vec{y})\}_{k=1}^{K_{PC}}$

– orthogonality also results in some sparsity in the left-hand side that may be taken advantage of when using iterative linear system solution methods

- for example, whenever $k + k' \neq k''$ (and for similar situations involving reversal of indices), the summand on the left-hand side vanishes

- Determining the PC-approximations of the data functions a and f may be costly since one has to determine a different expansion for every spatial quadrature point used in the finite element spatial discretization
  - of course, if the data is independent of $\mathbf{x}$, then only one expansion for each data function is needed

- We again point out that the economies resulting from the use of PC-expansions of the data functions are realizable only for linear SPDEs

KL-expansions of random data fields

- Now, suppose that the data functions a and f are Gaussian correlated random fields
  - then, we may determine the approximate KL-expansions
  $$a(\mathbf{x};\vec{y}) \approx \sum_{n=1}^{N} \sqrt{\lambda_n} a_n(\mathbf{x}) y_n$$
  and
  $$f(\mathbf{x};\vec{y}) \approx \sum_{n=1}^{N} \sqrt{\sigma_n} f_n(\mathbf{x}) y_n,$$
  - $\{\lambda_n, a_n(\mathbf{x})\}_{n=1}^{\infty}$ and $\{\sigma_n, f_n(\mathbf{x})\}_{n=1}^{\infty}$ are the eigenpairs of the covariance functions for a and f, respectively
  - recall that we have to assume (spherical) Gaussian variables since otherwise $\vec{y}$ is not a set of independent parameters

– substituting into the PC-discretization of the linear SPDE results in

$$\sum_{j=1}^{J}\sum_{k=1}^{K_{PC}} c_{jk}\sum_{n=1}^{N}\sqrt{\lambda_n}\left(\int_{\mathcal{D}} a_n(\mathbf{x})S\Big(\phi_j(\mathbf{x})\Big)T\Big(\phi_{j'}(\mathbf{x})\Big)\,d\mathbf{x}\right)$$
$$\left(\int_{\Gamma} y_n\rho(\vec{y})\Psi_k(\vec{y})\Psi_{k'}(\vec{y})\,d\vec{y}\right)$$
$$=\sum_{n=1}^{N}\sqrt{\sigma_n}\left(\int_{\mathcal{D}} f_n(\mathbf{x})\phi_{j'}(\mathbf{x})\,d\mathbf{x}\right)\left(\int_{\Gamma} y_n\rho(\vec{y})\Psi_{k'}(\vec{y})\,d\vec{y}\right)$$

- Doubly orthogonal polynomials can be constructed[†] such that

$$\int_{\Gamma}\Psi_k(\vec{y})\Psi_{k'}(\vec{y})\rho(\vec{y})\,d\vec{y}=0 \qquad \text{and} \qquad \int_{\Gamma}\vec{y}\,\Psi_k(\vec{y})\Psi_{k'}(\vec{y})\rho(\vec{y})\,d\vec{y}=0$$

whenever $k \neq k'$

[†]The construction involves solving an eigenvalue problem for each polynomial

- As a result, the probabilistic and spatial degrees of freedom uncouple
  - one can solve for the c_{ij}'s by solving K_{PC} deterministic finite element problems of size J instead of the single problem of size JK_{PC}
- We again point out that the economies resulting from the use of KL-expansions of the data random fields are realizable only for linear SPDEs
- Moreover, even for linear SPDEs, they are only possible for Gaussian random fields since it is only in this case that the KL expansions are linear in independent random parameters
- This should be contrasted with stochastic collocation methods and the non-intrusive polynomial chaos methods for which the uncoupling of the parameter and spatial degrees of freedom occurs for general, nonlinear SPDEs
  - for stochastic collocation methods, the uncoupling also occurs for general, non-Gaussian probability distributions

OPTIMAL CONTROL PROBLEMS FOR STOCHASTIC PARTIAL DIFFERENTIAL EQUATIONS

Optimization problems

- The state system

$$\begin{aligned} -\nabla\cdot\big(\kappa(\omega,\mathbf{x})\nabla u(\omega,\mathbf{x})\big) &= f(\omega,\mathbf{x}) && \text{in } \Omega\times D \\ u(\omega,\mathbf{x}) &= 0 && \text{on } \Omega\times\partial D \end{aligned}$$

  - ω is an elementary event in a probability space Ω
  - $\mathbf{x}$ is a point in the spatial domain D
  - $\kappa(\omega,\mathbf{x})$ and $f(\omega,\mathbf{x})$ are correlated random fields
  - the solution $u(\omega,\mathbf{x})$ is also a random field

- Optimal control problem
  - $\kappa(\omega, \mathbf{x})$ is given
  - $f(\omega, \mathbf{x})$ to be determined
  - given target function $\widehat{u}(\omega, \mathbf{x})$ may be deterministic or may be a random field
  - cost functional ($\mathsf{E}(\cdot)$ denotes the expected value)

$$\mathcal{F}(u, f; \widehat{u}) = \mathsf{E}\Big(\|u(\omega, \cdot) - \widehat{u}(\omega, \cdot)\|^2_{L^2(D)} + \alpha \|f(\omega, \cdot)\|^2_{L^2(D)} \Big)$$

$\Longrightarrow$

find a state u and a control f such that $\mathcal{F}(u, f; \widehat{u})$ is minimized subject to the state system being satisfied

- Parameter identification problem
  - $f(\omega, \mathbf{x})$ is given
  - $\kappa(\omega, \mathbf{x})$ to be determined
  - given target function $\widehat{u}(\omega, \mathbf{x})$ may be deterministic or may be a random field
  - cost functional

$$\mathcal{K}(u, \kappa; \widehat{u}) = \mathsf{E}\Big(\|u(\omega, \cdot) - \widehat{u}(\omega, \cdot)\|^2_{L^2(D)} + \beta \|\nabla \kappa(\omega, \cdot)\|^2_{L^2(D)} \Big)$$

$\Longrightarrow$

find a state u and a coefficient function κ such that $\mathcal{K}(u, \kappa; \widehat{u})$ is minimized subject to the state system being satisfied

Results

- Existence of optimal solutions
- Existence of Lagrange multipliers
- Derivation of optimality system
  - the adjoint or co-state system

$$\begin{aligned} -\nabla\cdot\big(\kappa(\omega,\mathbf{x})\nabla\xi(\omega,\mathbf{x})\big) &= -\big(u(\omega,\mathbf{x})-\widehat{u}(\omega,\mathbf{x})\big) && \text{in } \Omega\times D \\ \xi(\omega,\mathbf{x}) &= 0 && \text{on } \Omega\times\partial D \end{aligned}$$

  - optimality condition

$$\mathsf{E}\big(-\beta\Delta\kappa+\nabla u\cdot\nabla\xi\big)=0$$

- Discretization of noise so that κ, f, $\widehat{u}$, and u depend on a parameter vector $\vec{y}(\omega) = (y_1(\omega), \ldots, y_N(\omega))^T$
  - these parameters may be "knobs" in an experiment
  - alternately, they could result from an approximation, e.g., a truncated Karhunen-Loevy expansion, of a correlated random field
- finite element analyses of stochastic collocation method (in progress)
  - isotropic and anisotropic Smolyak sparse grids are used as collocation points
- development of gradient method to effect optimization

Computational results

- choose target $\widehat{u} = x(1-x^2) + \sum_{i=1}^{N} \sin\big(\frac{n\pi x}{L}\big) y_n(\omega)$

- choose optimal $\kappa = (1+x^3) + \sum_{i=1}^{N} \cos\big(\frac{n\pi x}{L}\big) y_n(\omega)$

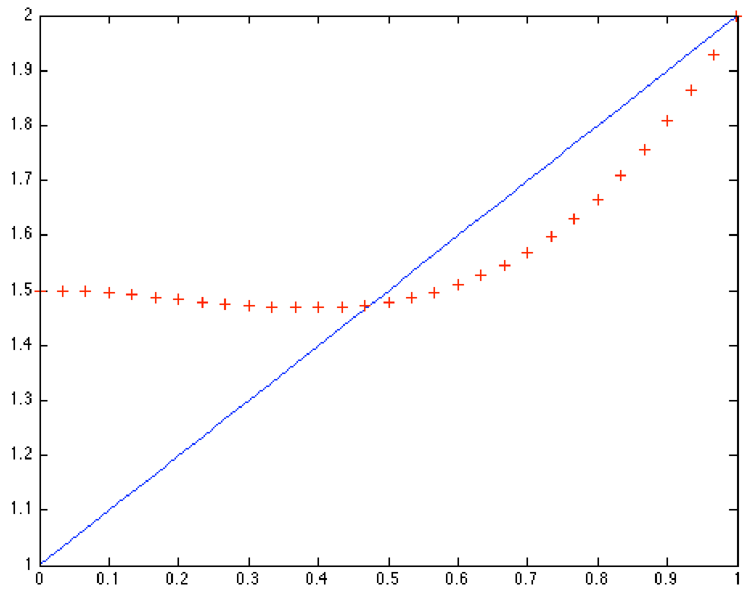
- set $f = -\nabla \cdot \big(\kappa \nabla \widehat{u}\big)$

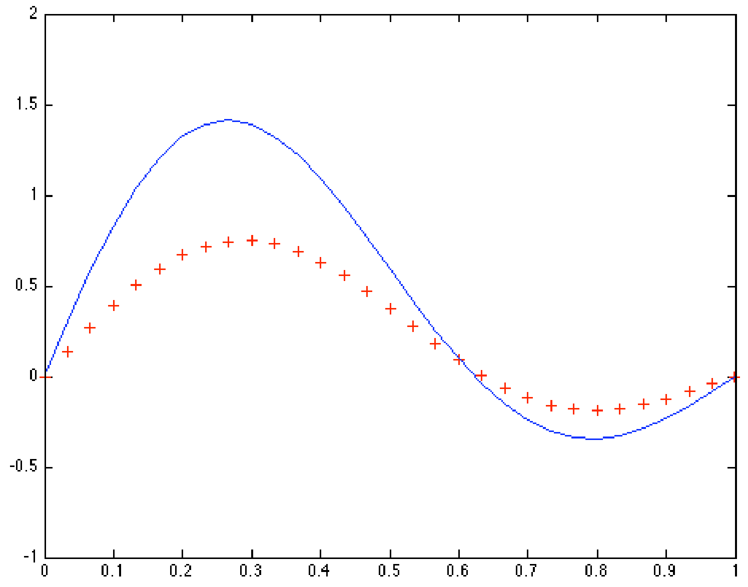
- choose initial $\kappa = 1 + x$

- assume y_i uniform on $[-1, 1]$ with $\mathsf{E}(y_i) = 0$ and $\mathsf{E}(y_i y_j) = \delta_{ij}$

$\Longrightarrow$

given random f and $\widehat{u}$, identify the expectation of both the control $\mathsf{E}(\kappa)$ and the state $\mathsf{E}(u)$ and compare with the exact statistical quantities

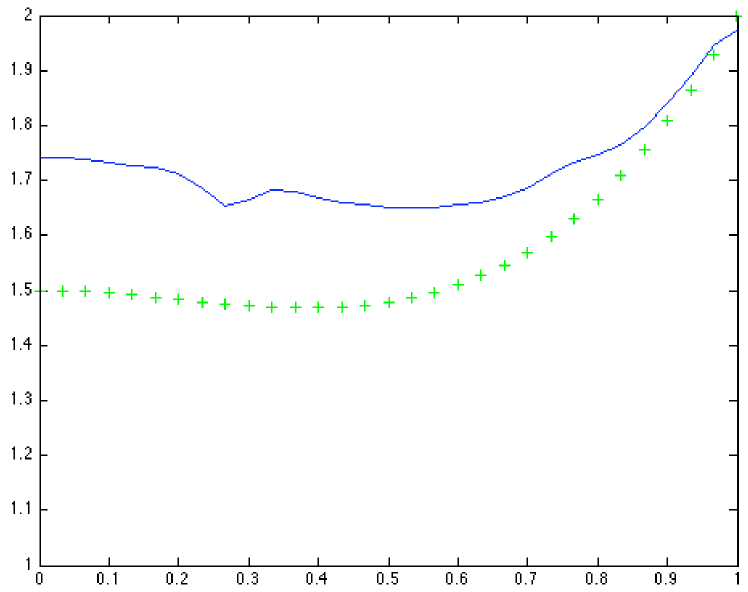
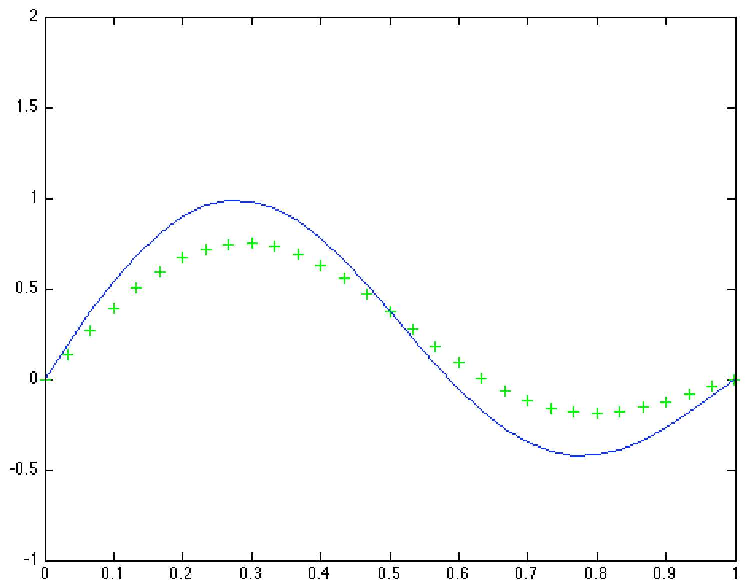




Left: expected value of initial (blue) and target (red) coefficient κ

Right: expected value of initial and target solution u

Number of random variables $= N = 1$

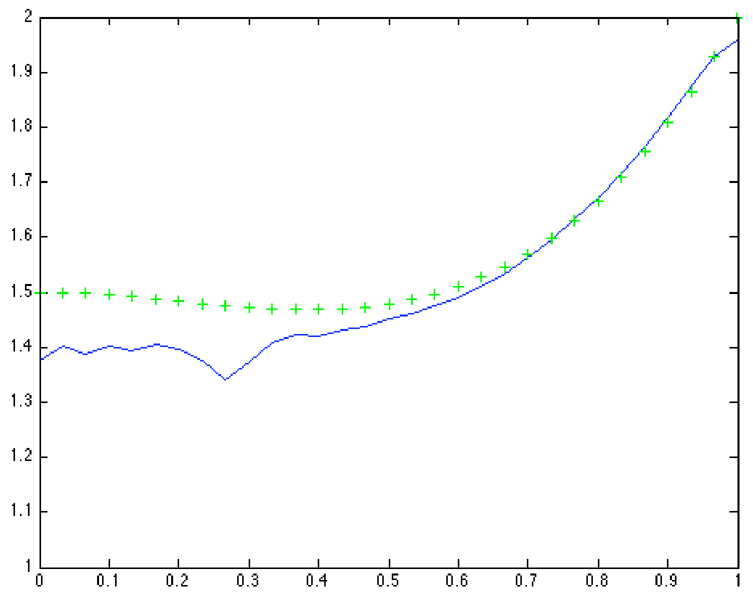



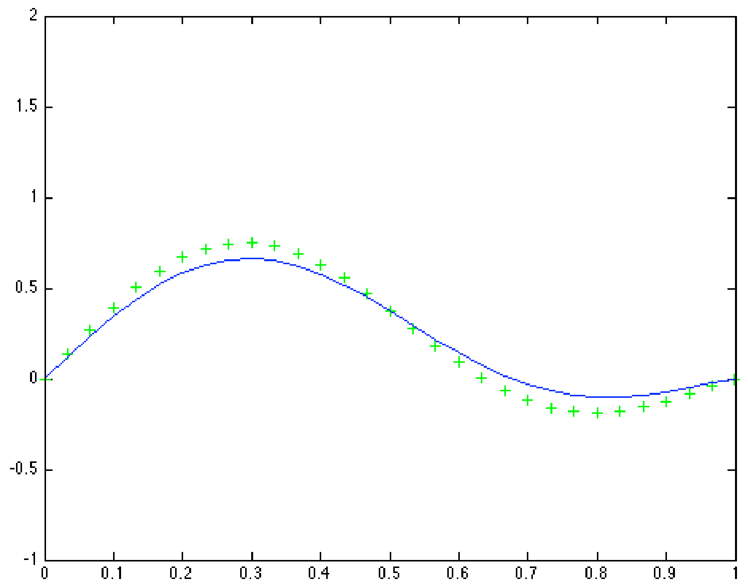
Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 1$

Number of Monte Carlo samples $= M = 1$



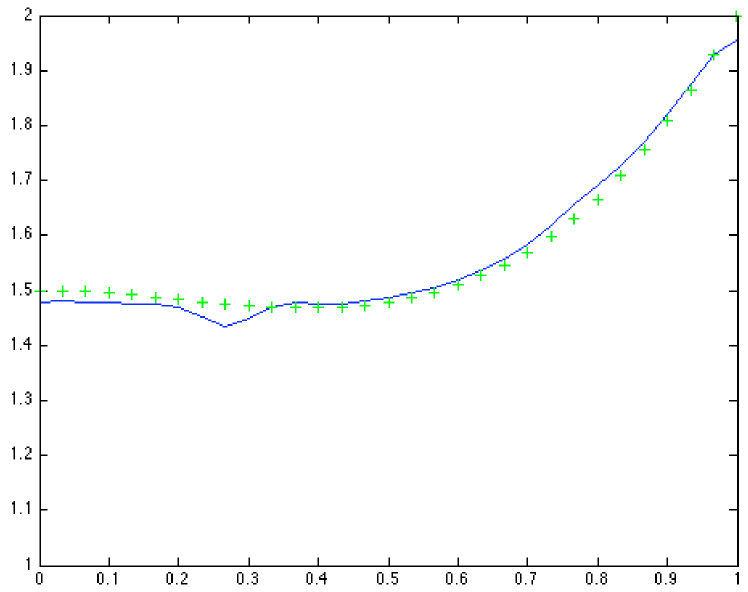


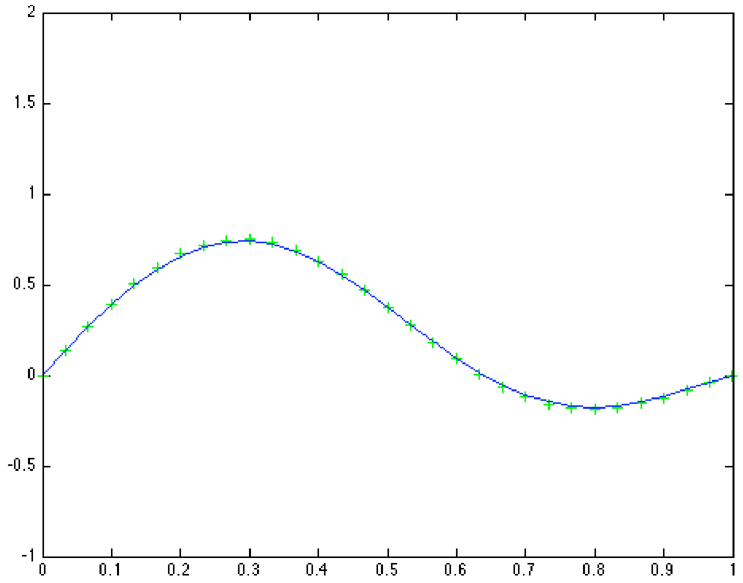
Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 1$

Number of Monte Carlo samples $= M = 10$



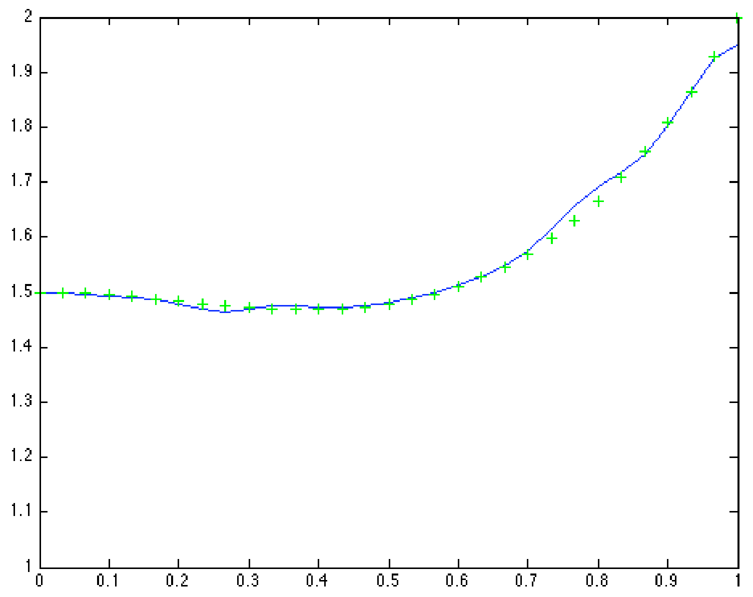


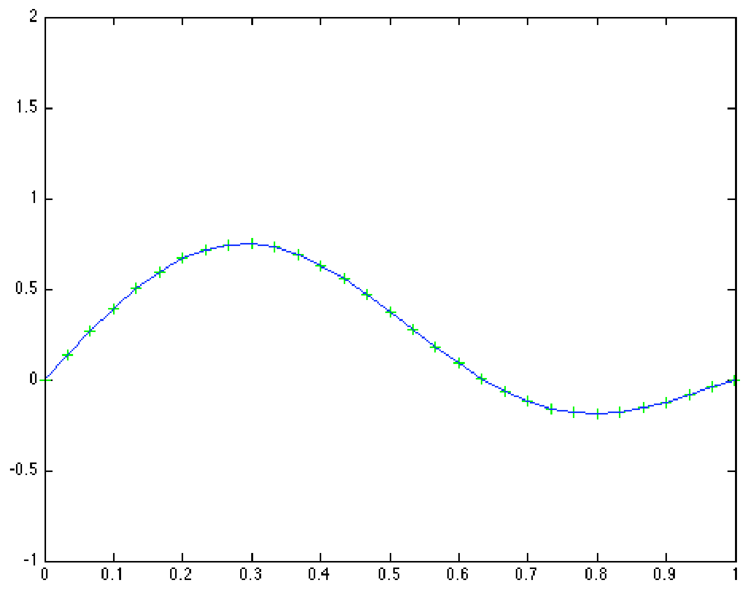
Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 1$

Number of Monte Carlo samples $= M = 100$



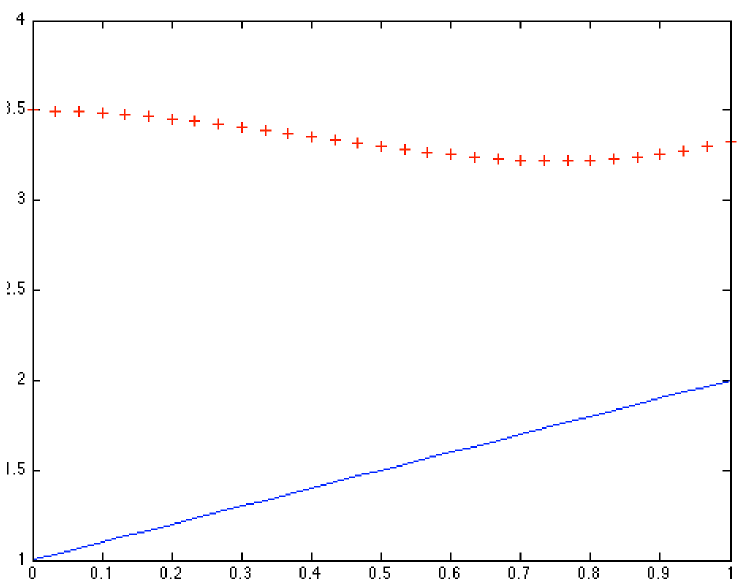
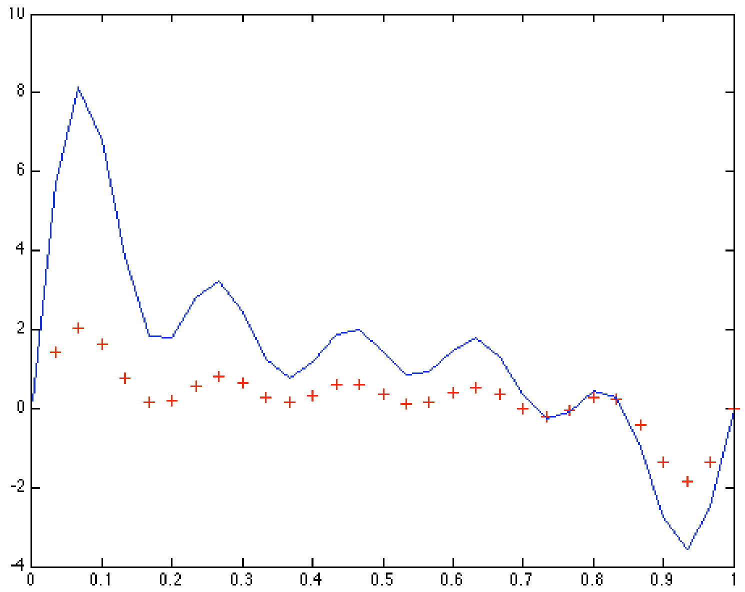


Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 1$

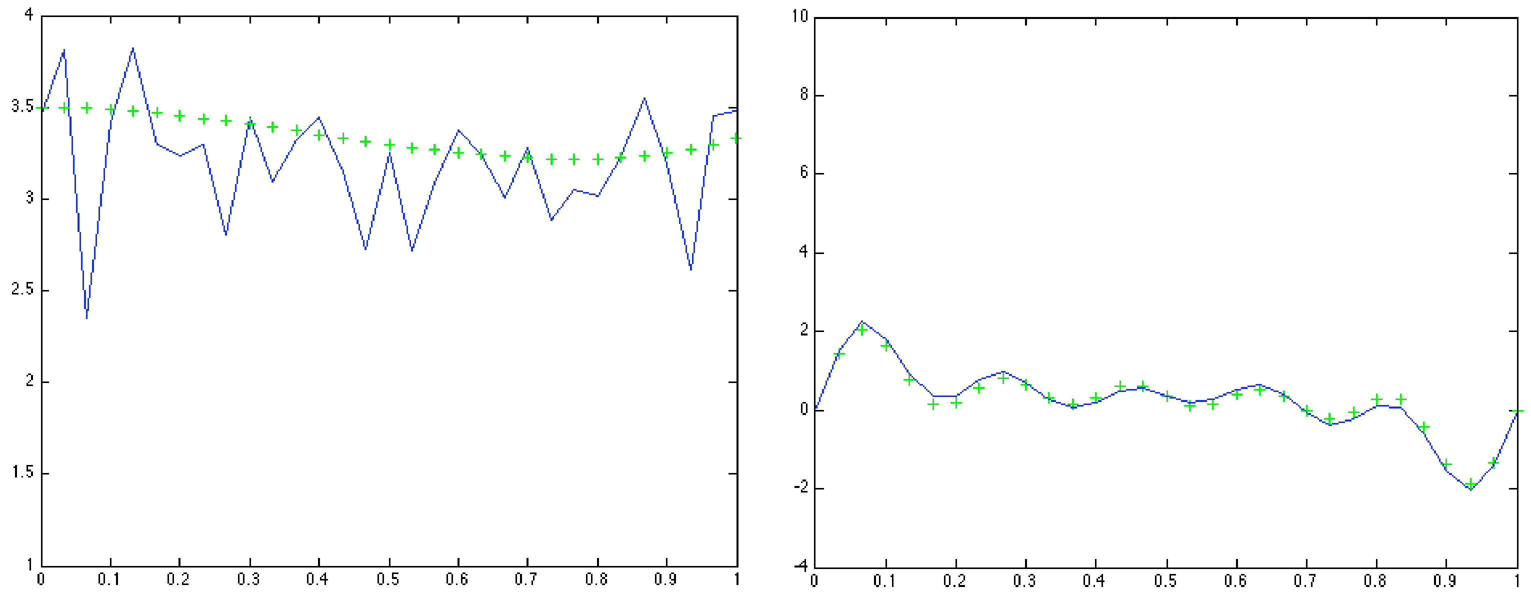
Number of anisotropic Smolyak collocation points $= M = 1$

Left: expected value of initial (blue) and target (red) coefficient κ

Right: expected value of initial and target solution u

Number of random variables $= N = 5$

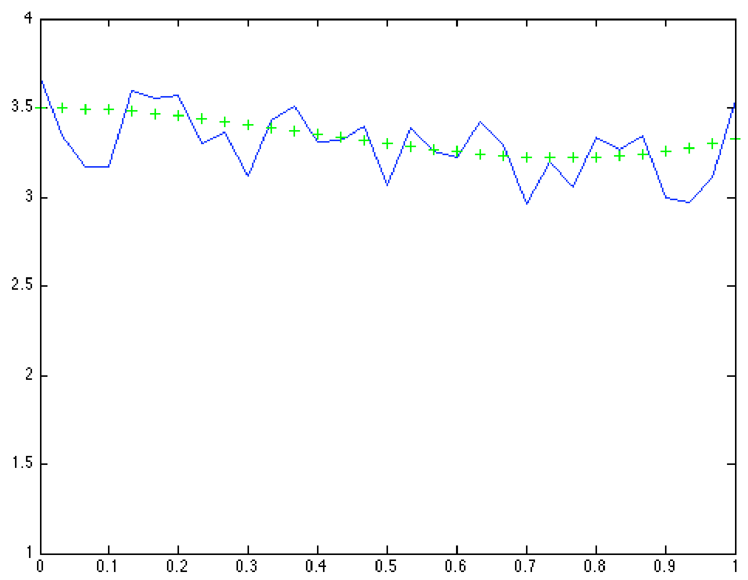
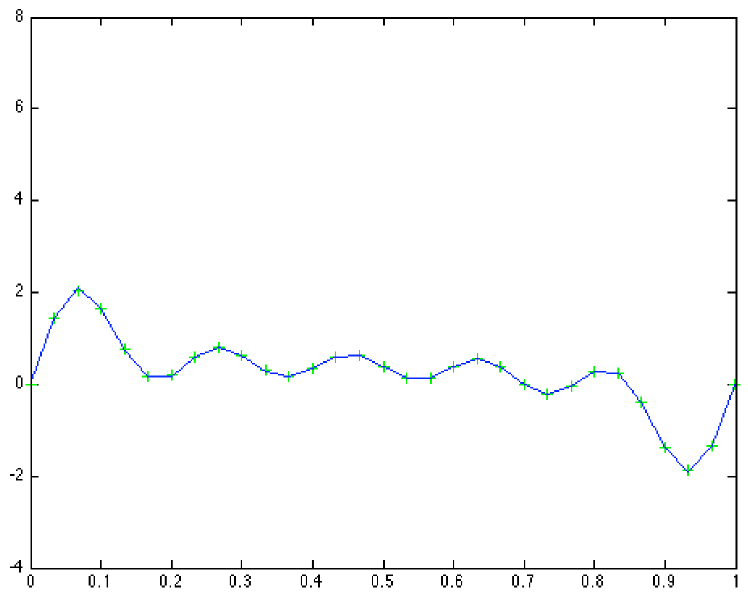


Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 5$

Number of Monte Carlo samples $= M = 11$

Left: expected value of optimal and target coefficient κ

Right: expected value of optimal and target solution u

Number of random variables $= N = 5$

Number of anisotropic Smolyak collocation points $= M = 11$

N	MC	AS
5	7e+03	801
10	9e+06	1581
20	8e+09	11561

For N random parameters, the number of Monte Carlo samples and the number of anisotropic Smolyak collocation points required to reduce the original error in the expected values of both the solution u and coefficient κ by a factor of 10^6