Polynomial Chaos Expansions for Uncertainty Quantification

AICES EU Regional School 2016 - Part 2

Ryan G. McClarren

Texas A&M University

Section 1

2-D Poisson Equation Example

- The examples we have seen so far have been functions that have been simple to evaluate.
- In those examples, there was no benefit to minimizing the number of function evaluations.
- For a more expensive evaluation, we solve the 2-D Poisson equation with Dirichlet boundary conditions:

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) u(x,y) = -q(x,y). \tag{1}$$

$$u(1,y) = u(x,1) = u(-1,y) = u(x,-1) = 0. \tag{2}$$

- The source q will be a Gaussian in space with an uncertain center in y:

$$q(x,y) = \exp\left[-x^2 - (y-\omega)^2\right]. \tag{3}$$

2-D Poisson Equation Example

- The center of the Gaussian in the y-coördinate will be a uniform random variable in the range $[-0.25, 0.25]$ (i.e., $\omega \sim \mathcal{U}(-0.25, 0.25)$).
- We are interested in the integral over a quarter of the domain. Our quantity of interest is therefore

$$g(\omega) = \int_0^1 dx \int_0^1 dy\, u(x,y;\omega). \tag{4}$$

- The notation $u(x,y;\omega)$ denotes that the solution depends on the center of Gaussian ω.

2-D Poisson Equation Example

- We will estimate the Legendre expansion coefficients using Gauss-Legendre quadrature.
- Using an $n = 2$ quadrature rule we would estimate the coefficients as

$$c_n \approx \frac{2n+1}{2}\left(g\left(-\frac{1}{4\sqrt{3}}\right)P_n\left(-\frac{1}{4\sqrt{3}}\right)+g\left(\frac{1}{4\sqrt{3}}\right)P_n\left(\frac{1}{4\sqrt{3}}\right)\right). \tag{5}$$

- To compute the c_n in this case will require solving Poisson's equation twice, each time with different sources, and computing the integral in Eq. (4).
- We use Mathematica's `NDSolve` function with these two values of ω to get

$$g\left(-\frac{1}{4\sqrt{3}}\right) = 0.381378, \qquad g\left(\frac{1}{4\sqrt{3}}\right) = 0.381378.$$

- The mean of the function is

$$c_0 \approx \frac{1}{2}\left[g\left(-\frac{1}{4\sqrt{3}}\right)+g\left(\frac{1}{4\sqrt{3}}\right)\right] = 0.381378. \tag{6}$$

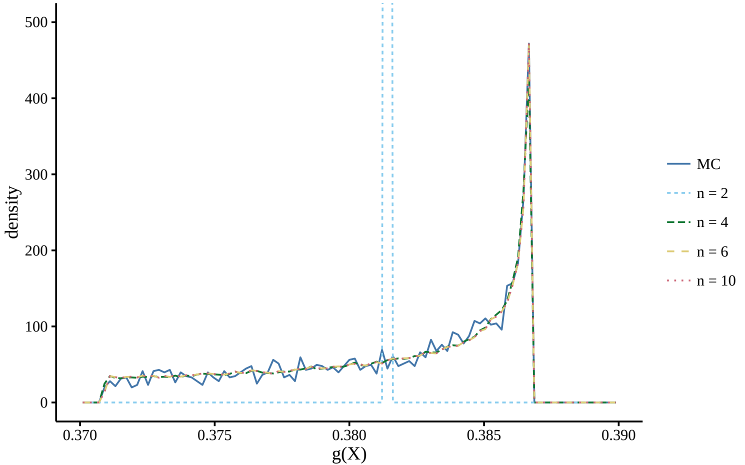
2-D Poisson Equation Example: Coefficients

n	c_0	c_1	c_2	c_3	c_4	c_5
1	0.386712	0.000000	-0.966780	0.000000	1.305153	0.000000
2	0.381378	0.000000	-0.000000	-0.000000	-1.334823	-0.000000
3	0.381406	-0.000000	-0.010613	-0.000000	0.014327	0.000000
4	0.381406	-0.000000	-0.010559	0.000000	-0.000000	0.000000
5	0.381406	0.000000	-0.010559	0.000000	0.000071	-0.000000
6	0.381406	-0.000000	-0.010559	0.000000	0.000071	-0.000000
7	0.381406	-0.000000	-0.010559	0.000000	0.000071	-0.000000
8	0.381409	0.000000	-0.010567	-0.000000	0.000079	0.000000
9	0.381406	0.000000	-0.010559	-0.000000	0.000071	-0.000000
10	0.381406	0.000000	-0.010559	-0.000000	0.000071	-0.000000

1: The convergence of the first six coefficients in the 2-D Poisson equation example as a function of the number of Gauss-Legendre quadrature points used.

- Note that in the best case, we could only hope for a quadrature rule with n points to integrate up to c_{2n-1} accurately, and that this would only be the case if g we are a constant function.
- From this table it seems that the integrals are accurate (though not exact) up to c_n for an n point quadrature rule, for $n > 2$.

2-D Poisson Equation Example



PDF of the random variable $g(\omega) = \int_0^1 dx \int_0^1 dy\, u(x,y;\omega)$, where $\omega \sim \mathcal{U}(-0.25, 0.25)$ and u is the solution to Eq. (1), using several different Gauss-Legendre quadrature rules and a Monte Carlo simulation using 3×10^3 numerical solutions of Poisson's equation.

Section 2

Expansions in More than One Dimension

- It is likely that in a realistic problem there will be several sources of uncertainty and several uncertain parameters.
- It may also be possible that the different parameters may have different types of distributions.
- Consider a generic function of d random variables, θ_i, with an expansion given by

$$g(\theta_1, \ldots, \theta_d) = \sum_{l_1=0}^{\infty} \cdots \sum_{l_d=0}^{\infty} c_{l_1,\ldots,l_d} \mathfrak{P}_{l_1,\ldots,l_d}(\theta_1, \ldots, \theta_d). \quad (7)$$

- Here $\mathfrak{P}_{l_1,\ldots,l_d}(\theta_1, \ldots, \theta_d)$ is a product of the d orthogonal polynomials,

$$\mathfrak{P}_{l_1,\ldots,l_d}(\theta_1, \ldots, \theta_d) = \prod_{i=1}^{d} P_{l_i}(\theta_i), \quad (8)$$

Expansions in More than One Dimension

- The expansion coefficients are

$$c_{l_1,\dots,l_d} = \int_{D_1} d\theta_1 \cdots \int_{D_d} d\theta_d \, g(\theta_1,\dots,\theta_d) \mathfrak{P}_{l_1,\dots,l_d}(\theta_1,\dots,\theta_d) \mathfrak{w}(\theta_1,\dots,\theta_d), \tag{9}$$

- $\mathfrak{w}(\theta_1,\dots,\theta_d)$ is the product of the weight functions for the d bases.
- If the sum is truncated at degree N polynomials then there will be $(1+N)^d$ terms in the expansion.

Expansions in More than One Dimension

- Consider the function $g = \cos(\theta_1)\cos(\theta_2)$ and $\theta_i \sim \mathcal{U}(0, 2\pi)$
- A second-order expansion would have the form

$$\begin{aligned} g(\theta_1, \theta_2) = c_{0,0} + c_{1,0}P_1(\pi\theta_1 + \pi) + c_{0,1}P_1(\pi\theta_2 + \pi) + \\ c_{2,0}P_2(\pi\theta_1 + \pi) + c_{0,2}P_2(\pi\theta_2 + \pi) + \\ c_{1,1}P_1(\pi\theta_1 + \pi)P_1(\pi\theta_2 + \pi) + \\ c_{2,1}P_2(\pi\theta_1 + \pi)P_1(\pi\theta_2 + \pi) + \\ c_{1,2}P_1(\pi\theta_1 + \pi)P_2(\pi\theta_2 + \pi) + \\ c_{2,2}P_2(\pi\theta_1 + \pi)P_2(\pi\theta_2 + \pi). \end{aligned} \tag{10}$$

Tensor-Product Quadrature

- To compute the expansion coefficients we can use what is known as a tensor-product quadrature set.
- Here we take a 1-D quadrature rule with n points and weights given by $\{w_i, x_i\}$ for $i = 1 \dots n$, that we denote as Q_n so that

$$Q_n f(x) = \sum_{l=1}^{n} w_l f(x_l), \tag{11}$$

- Apply it over all dimensions as

$$Q_n^{(d)} g(\theta_1, \dots, \theta_d) = \sum_{l_1=1}^{n} \cdots \sum_{l_d=1}^{n} w_{l_1} \cdots w_{l_d} g(\theta_{1l_1}, \dots, \theta_{dl_d}), \tag{12}$$

where θ_{i,l_j} is the ith input evaluated at its jth point in the quadrature set.

Tensor-Product Quadrature

- It is convenient to write $Q^{(d)}$ as a tensor product of 1-D quadrature rules. We define a tensor product of two quadrature rules as

$$Q_n \otimes Q_m = \{\{w_i w_j, (x_i, x_j)\} : i = 1 \dots n, j = 1 \dots m\}. \tag{13}$$

  Therefore, we can write a tensor-product quadrature comprised of n point quadratures as

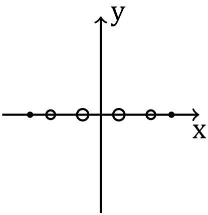
$$Q_n^{(d)} g(\theta_1, \dots, \theta_d) = (Q_n^{(1)} \otimes \cdots \otimes Q_n^{(d)}) g. \tag{14}$$

- We could in principle have each dimension have a different number of quadrature points, and in many cases this will make the calculation more efficient.

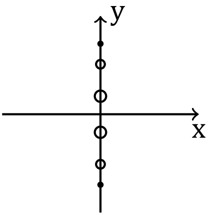
Tensor-Product Quadrature

- The number of quadrature points scales geometrically with d.
- This is the so-called curse of dimensionality because the number of function evaluations needed explodes as d gets larger.
- Using a two-point quadrature rule, when $d = 26$, requires one simulation for every person in Germany.
- Even worse $d = 26$ requires a mole (6×10^{23}) of calculations for the $n = 8$ rule.
- In a full-scale engineering system, 26 uncertain parameters is not out of the question.

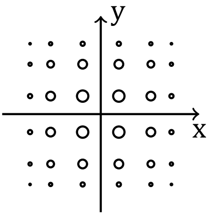
Tensor-Product Quadrature



1-D quadrature in x



1-D quadrature in y



Tensor product quadrature

Tensor-Product Quadrature

Two things are evident in previous figure:

- The weights are much larger in the middle of domain, and
- The points are more densely packed near the corners.
- These effects are even more pronounced as the number points in the quadrature set goes up.

Section 3

Black-Scholes Model

- We will look at the solution to the Black-Scholes partial differential equation for the value of a call option.
- A call option gives the holder the ability to purchase a stock at a given price, called the strike price, at a given future date.
- The value of the option is a function of the current price of the stock (S), the strike price (K), the time to expiration in years (T), the risk-free interest rate (r), the dividend rate the stock pays q, and the volatility of the stock (σ).
- Three of these, r, q, and σ are uncertain parameters.
- The Black-Scholes model is based on assuming that the stock price follows geometric Brownian motion.

Black-Scholes Model

- The solution for the price, p, of an option from the Black-Scholes model can be given by

$$p = e^{-rT}\left(F\Phi(v_1) - K\Phi(v_2)\right), \tag{15}$$

where

$$F = Se^{(r-q)T}, \tag{16}$$

$$v_1 = \frac{\log\frac{S}{K} + (r - q + \frac{1}{2}\sigma^2)T}{T\sqrt{T}}, \qquad v_2 = v_1 - \sigma\sqrt{T}, \tag{17}$$

and $\Phi(z)$ is the standard normal CDF function.

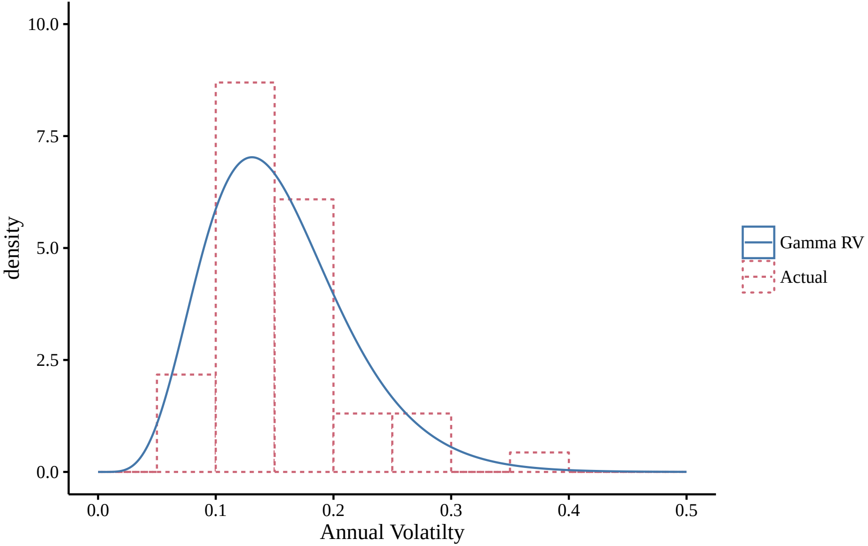
Black-Scholes Example

- We are interested in calculating the current value of a call option for stock in the Coca-Cola company, ticker symbol KO.
- On 15 August 2016, KO was trading at \$44.15.
- We will consider a call with strike price of $K = \$44$.
- The option expiration is 158 days away ($T = 0.432877$).
- This option was trading at \$1.46 on that day, we want to know how that compares to the model.
- We need to estimate the random variables, r, q, and σ.
- For the interest rate, r, we will use the benchmark LIBOR 30-day interest rate with a Gamma distribution $r = 0.0048x$ with $X \sim \mathcal{G}(0,1)$.
  - This distribution had a mean of the current rate (0.48%).
- For the dividend rate we will use a uniform distribution so that $Q \sim \mathcal{U}(0.025, 0.045)$.

Black-Scholes Example

- For the volatility, σ, we look at the actual annual standard deviations of daily returns for the years from 1970 to 2015.
- The histogram of these 45 volatilities is shown on the next slide,
- We compute a Gamma distribution that matches the mean and variance of the observations, $\Sigma \sim \mathcal{G}(5.46636, 41.8142)$.

Black-Scholes Example: Coca-Cola Volatility



Black-Scholes Example: Expansion

- The expansion of $p(X, D, \Sigma)$ will have the form

$$p(X, Q, \Sigma) = \sum_{l_x=0}^{\infty} \sum_{l_d=0}^{\infty} \sum_{l_\sigma=0}^{\infty} c_{l_x l_d l_\sigma} L_{l_x}^{(0)}(x) P_{l_d}\left(\frac{2d - 0.7}{0.2}\right) L_{l_\sigma}^{(5.46636)}(41.8142\sigma). \tag{18}$$

- From this equation, we can compute the mean of the distribution, c_{000} as

$$\bar{p} = c_{000} = \int_0^{\infty} dx \int_{0.025}^{0.045} dq \int_0^{\infty} dz\, p\left(x, q, \frac{z}{41.8142}\right) \frac{z^{5.46636}}{\Gamma(6.46636)} e^{-x-z} \left(\frac{1}{0.02}\right)$$
$$\approx 1.56662. \tag{19}$$

- Note that this is slightly higher than the price the option is trading at, \$1.46.

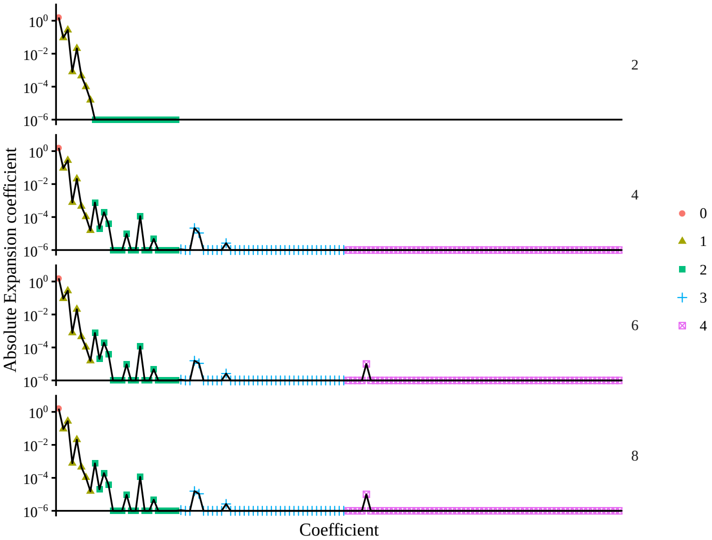
Black-Scholes Example: Fourth-Order Expansion

- Because the price of the option is a well-behaved function, we will expand p with polynomial degree up to order four:

$$p(X, Q, \Sigma) = \sum_{l_x=0}^{4} \sum_{l_d=0}^{4} \sum_{l_\sigma=0}^{4} c_{l_x l_d l_\sigma} L_{l_x}^{(0)}(x) P_{l_d}\left(\frac{2d - 0.7}{0.2}\right) L_{l_\sigma}^{(5.46636)}(41.8142\sigma). \quad (20)$$

- Such an expansion will have $5^3 = 125$ terms.
- Using tensor-product Gauss quadrature—Gauss-Laguerre in x and σ, Gauss-Legendre in q—we can estimate these coefficients.

Black-Scholes Example: Coefficient Estimation



Black-Scholes Example: Coefficient Estimation

- In the previous figure the results from these calculations with various numbers of points in the 1-D quadrature rules that comprise the tensor-product quadrature are shown.
- This figure indicates the maximum single polynomial degree in each point using a color/shape.
- Here we only show coefficients with a magnitude larger than 10^{-6},
- For the $n = 2$ rules we do not show any coefficients corresponding to degree three or four polynomials.

Black-Scholes Example: Coefficient Estimation

- We see that the $n = 2$ quadrature rule does a good job of estimating the low-order, large-magnitude coefficients.
- This indicates that most of the variation in the distribution can be captured using only $2^3 = 8$ evaluations of the function.
- The higher-order coefficients have a smaller magnitude and can be captured using $n = 4$ rules,
- The largest, significant coefficient c_{004} or the coefficient for a quartic in volatility can be captured using $n = 6$
  - A total of $6^3 = 216$ function evaluations.

Black-Scholes Example: Variance Estimates

- One way to compare the quadrature rules is to look at the convergence of the variance. The variance in the price is

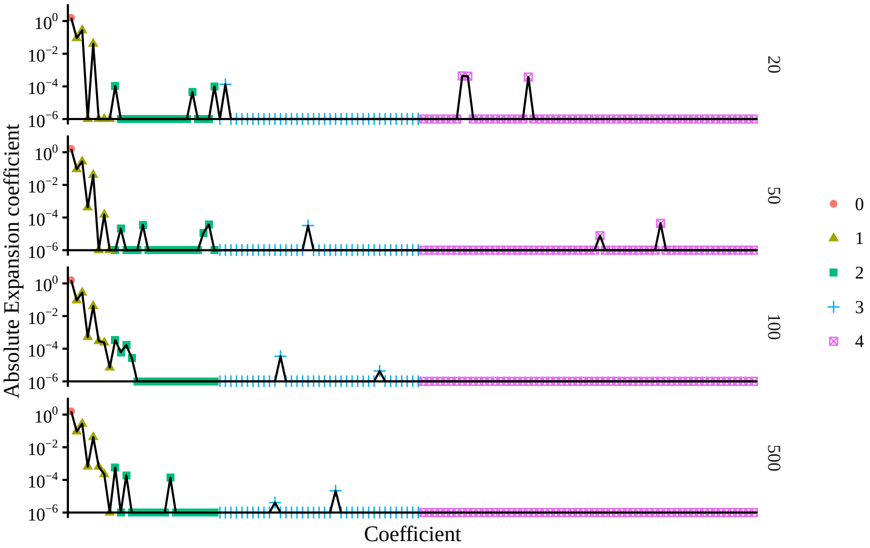
$$\mathrm{Var}(P) = \sum_{l_x=1}^{\infty} \sum_{l_d=1}^{\infty} \sum_{l_\sigma=1}^{\infty} \frac{\Gamma(l_x+1)\Gamma(l_\sigma+6.46636)}{l_x! l_\sigma! \Gamma(6.46636)(2l_d+1)} c^2_{l_x l_d l_\sigma}. \qquad (21)$$

- The results for this calculation using the expansion coefficients are shown below. This table indicates that the $n = 2$ coefficients estimate the variance to three digits of accuracy.

n	$\mathrm{Var}(P)$
2	0.486085
4	0.486321
6	0.486321
8	0.486321

2: The convergence of the variance in the option price as a function of the quadrature rule used.

Black-Scholes Example: Coefficient Estimation



Black-Scholes Example: Conclusions

From this example, several things are evident.

- With a smoothly varying function, the expansion order required to estimate the distribution of the quantity of interest, and the number of function evaluations needed are small.
- The results also indicate that of the many coefficients possible in a high-order expansion will be negligible.
- Next, we will investigate how to take advantage of this structure.

Section 4

Maximum-Order Expansions

- The explosion of terms in multi-dimensional expansions comes, in part, from the cross-terms that appear in the expansion.
- For example, in a fourth-order expansion we end up with 4^d degree polynomials because the highest order terms in the series are a product of four d-degree polynomials.
- The tensor-product Gauss quadratures that we use to estimate the expansion can accurately integrate these polynomials.
- Nevertheless, these high-degree interactions (that is the product of several high-degree polynomials) are often unnecessary in the expansion (as we saw in the previous case).

Maximum-Order Expansions

- In such a scenario it can be useful to change the way that we expand the output in orthogonal polynomials.
- Instead of including combinations of polynomials up to a given degree, we look to include only polynomials up to a maximum degree:

$$g(\theta_1 \dots, \theta_d) \approx \sum_{l_1 + \dots + l_d < N} c_{l_1, \dots, l_d} \mathfrak{P}_{l_1, \dots, l_d}(\theta_1, \dots, \theta_d), \tag{22}$$

- With this expansion, we no longer need to integrate any polynomials of degree higher than N.
- Therefore, our tensor-product quadrature rule integrates higher-degree polynomials than we need.

Smolyak Sparse Quadrature

- For this situation we can use Smolyak sparse quadrature sets.
- These rules construct quadrature points that do not grow as fast as product quadrature grids.
- To accomplish this, we combine quadrature rules to ensure that a polynomial of a given degree in any single dimension, but not products of polynomials of that degree, is exactly integrated.
- For a given value of ℓ, in d dimensions the quadrature rule is defined as

$$S_\ell^{(d)} f = \sum_{q=\ell-d}^{\ell-1} (-1)^{\ell-1-q} \binom{d-1}{\ell-1-q} \sum_{||\vec{k}||_1 = q+d} Q_{2^{k_1}-1} \otimes \cdots \otimes Q_{2^{k_d}-1} f, \tag{23}$$

where $||\vec{k}||_1 = \sum_{i=1}^{d} |k_i|$.

- Looking at this formula we see that the tensor products where the sum of the number of points in each dimension equals a constant are included.
- Note that the quadrature rule can have negative weights.

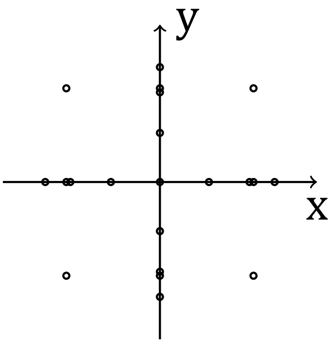
Smolyak Sparse Quadrature Example in 2-D

- To demonstrate how these rules work we will look at the quadrature rule with $\ell = 3$ and Gauss-Legendre quadrature.
- In this case we should have the a quadrature rule with up to $2^3 - 1$ points:

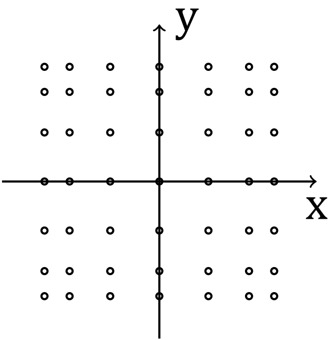
$$
\begin{aligned}
S_3^{(2)} f &= \sum_{q=1}^{2} (-1)^{2-q} \binom{1}{2-q} \sum_{\|\vec{k}\|_1 = q+2} Q_{2^{k_1}-1} \otimes Q_{2^{k_2}-1} f \\
&= -\sum_{\|\vec{k}\|_1 = 3} Q_{2^{k_1}-1} \otimes Q_{2^{k_2}-1} f + \sum_{\|\vec{k}\|_1 = 4} Q_{2^{k_1}-1} \otimes Q_{2^{k_2}-1} f \\
&= -(Q_1 \otimes Q_3) f - (Q_3 \otimes Q_1) f + (Q_3 \otimes Q_3) f + (Q_1 \otimes Q_7) f + (Q_7 \otimes Q_1) f
\end{aligned}
$$

- Counting up the total number of points in this rule there are 21 compared to 49 for the tensor product quadrature rule for $Q_7 \otimes Q_7$.
- The $(Q_1 \otimes Q_3)$ and $(Q_3 \otimes Q_1)$ rules are completely redundant with $(Q_3 \otimes Q_3)$ and the $(Q_1 \otimes Q_7)$ and $(Q_3 \otimes Q_1)$ rules share the origin with $(Q_3 \otimes Q_3)$.

Smolyak Sparse Quadrature Example in 2-D



$S_3^{(2)}$ rule



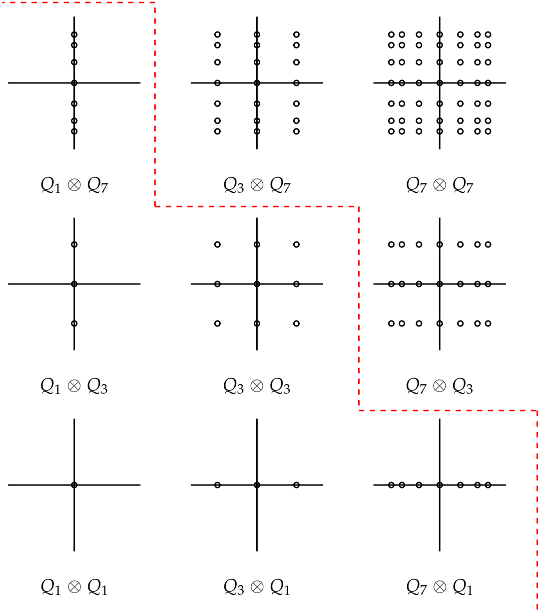
$Q_7 \otimes Q_7$ rule

2: Comparison of the Smolyak sparse quadrature rule of level $\ell = 3$ and the tensor-product rule comprised of 7-point Gauss-Legendre quadrature rules.

Smolyak Sparse Quadrature Example in 2-D

- Another way to show the construction of a 2-D Smolyak quadrature rule is to
  - Write all the quadrature rules up to order $2^\ell - 1$ in a tableau of tensor-product quadratures where
  - The number of points in the $\mathbf{x}$-direction increases from left to right, and
  - The number of points in the $\mathbf{y}$-direction increases from bottom to top.
- The Smolyak quadrature rule will be a linear combination of the tensor-product rules from the diagonal and below.
- This construction is shown on the next slide.

Smolyak Sparse Quadrature Construction



Smolyak Sparse Quadrature Example in 2-D

- Now that we have seen how the sparse grids work, we will discuss why they are constructed in the form that they are.
- As we have said, a product quadrature rule comprised of n points in 1-D, will integrate d-dimensional polynomials where *any single* component polynomial has degree less than or equal to $(2n - 1)$.
- The Smolyak construction is designed to integrate polynomials with a total degree of equal to $(2n - 1)$.
- This is shown in on the next slide for $n = 2$.
- It can be shown that the Smolyak sparse grid that is exact on polynomials of N in the one-dimensional quadrature rules will be exact on polynomials of total degree N for the multi-dimensional integral.

Smolyak Sparse Quadrature Construction

$$1$$

$$x \qquad y$$

$$x^2 \qquad xy \qquad y^2$$

$$x^3 \qquad x^2y \qquad xy^2 \qquad y^3$$

$$x^3y \qquad x^2y^2 \qquad xy^3$$

$$x^3y \qquad xy^3$$

$$x^3y^3$$

Smolyak Sparse Quadrature Example in 2-D

- The construction of the quadrature set will illuminate the origin of Eq. (23), in particular why there needs to be negatively-weighted points.
- Looking at the previous slide, to integrate the polynomials in the triangle we can think about it in terms of "adding" quadrature rules:

$$
\begin{pmatrix} & & 1 & & \\ & x & & y & \\ x^2 & & xy & & y^2 \end{pmatrix} =
$$

$$
\begin{pmatrix} & & 1 & & \\ & x & & & \\ x^2 & & & & \end{pmatrix} + \begin{pmatrix} & & 1 & & \\ & & & y & \\ & & & & y^2 \end{pmatrix} + \begin{pmatrix} & & 1 & & \\ & x & & y & \\ & & xy & & \end{pmatrix}
$$

$$
- \begin{pmatrix} & & 1 & & \\ & x & & & \\ & & & & \end{pmatrix} - \begin{pmatrix} & & 1 & & \\ & & & y & \\ & & & & \end{pmatrix} \tag{24}
$$

Smolyak Sparse Quadrature Example in 2-D

- Now that we have seen how the sparse grids work, we will discuss why they are constructed in the form that they are.
- As we have said, a product quadrature rule comprised of n points in 1-D, will integrate d-dimensional polynomials where *any single* component polynomial has degree less than or equal to $(2n - 1)$.
- The Smolyak construction is designed to integrate polynomials with a total degree of equal to $(2n - 1)$.
- This is shown in on the next slide for $n = 2$.
- It can be shown that the Smolyak sparse grid that is exact on polynomials of N in the one-dimensional quadrature rules will be exact on polynomials of total degree N for the multi-dimensional integral.

Section 5

Black-Scholes w/ Sparse Quadrature

- Turning back to our Black-Scholes example from before, we will construct a Smolyak sparse grid for this 3-D expansion.
- As we saw in before, a tensor-product quadrature comprised of six-point quadrature rules was able to capture the most important coefficients in the expansion.
- This rule has $6^3 = 216$ function evaluations.
- We will use the $\ell = 3$ Smolyak sparse grid to compute the coefficients in the expansion.

Black-Scholes w/ Sparse Quadrature

This quadrature rule can be calculated from

$$
\begin{aligned}
S_3^{(3)} f = & \sum_{q=0}^{2} (-1)^{2-q} \binom{2}{2-q} \sum_{\|\vec{k}\|_1 = q+3} Q^{(\sigma)}_{2^{k_1}-1} \otimes Q^{(x)}_{2^{k_2}-1} \otimes Q^{(z)}_{2^{k_3}-1} f \\
= & Q_1^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_1^{(z)} f - 2 \left[Q_3^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_1^{(z)} f + Q_1^{(\sigma)} \otimes Q_3^{(x)} \otimes Q_1^{(z)} f + Q_1^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_3^{(z)} f \right] \\
& + Q_7^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_1^{(z)} f + Q_1^{(\sigma)} \otimes Q_7^{(x)} \otimes Q_1^{(z)} f + Q_1^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_7^{(z)} f + \\
& + Q_3^{(\sigma)} \otimes Q_3^{(x)} \otimes Q_1^{(z)} f + Q_3^{(\sigma)} \otimes Q_1^{(x)} \otimes Q_3^{(z)} f + Q_1^{(\sigma)} \otimes Q_3^{(x)} \otimes Q_3^{(z)} f
\end{aligned}
$$

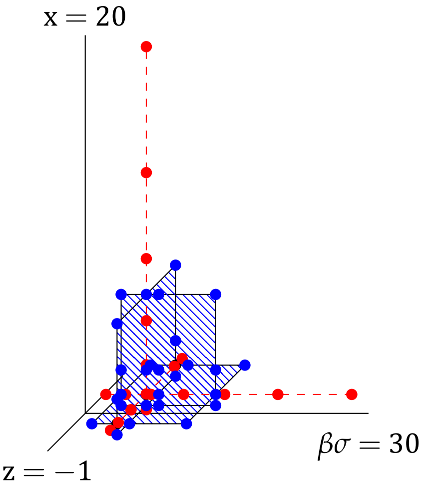
3-D Sparse Quadrature Rules

	$\beta\sigma$	w_σ	x	w_x	z	w_z
Q_1	6.466360	271.060701	1.000000	1.000000	0.000000	2.000000
	13.811184	13.236834	6.289945	0.010389	0.774597	0.555556
Q_3	7.787369	148.010162	2.294280	0.278518	0.000000	0.888889
	3.800528	109.813705	0.415775	0.711093	-0.774597	0.555556
	28.226889	0.000454	19.395728	0.000000	0.949108	0.129485
	20.399826	0.129138	12.734180	0.000016	0.741531	0.279705
	14.769642	4.663395	8.182153	0.001074	0.405845	0.381830
Q_7	10.417345	42.165053	4.900353	0.020634	0.000000	0.417959
	6.984121	116.015439	2.567877	0.147126	-0.405845	0.381830
	4.281556	93.279531	1.026665	0.421831	-0.741531	0.279705
	2.185142	14.807693	0.193044	0.409319	-0.949108	0.129485

3: The 1-D quadrature rules that comprise the sparse rule $S_3^{(3)}$.

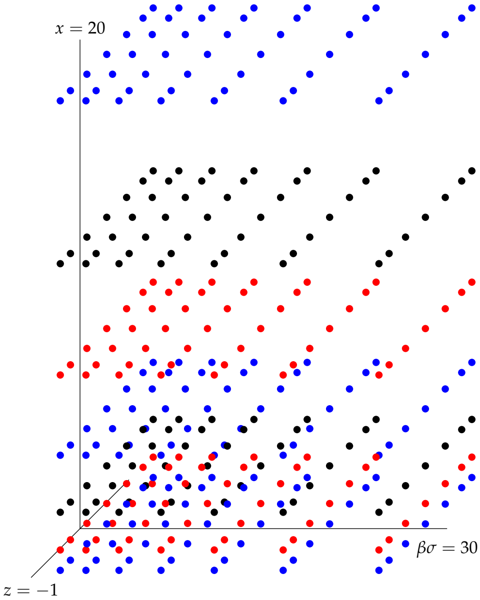
The nesting of points in the z direction leads to 7 redundant points and a total of 50 unique points in the $S_3^{(3)}$ set.

Black-Scholes w/ Sparse Quadrature Points

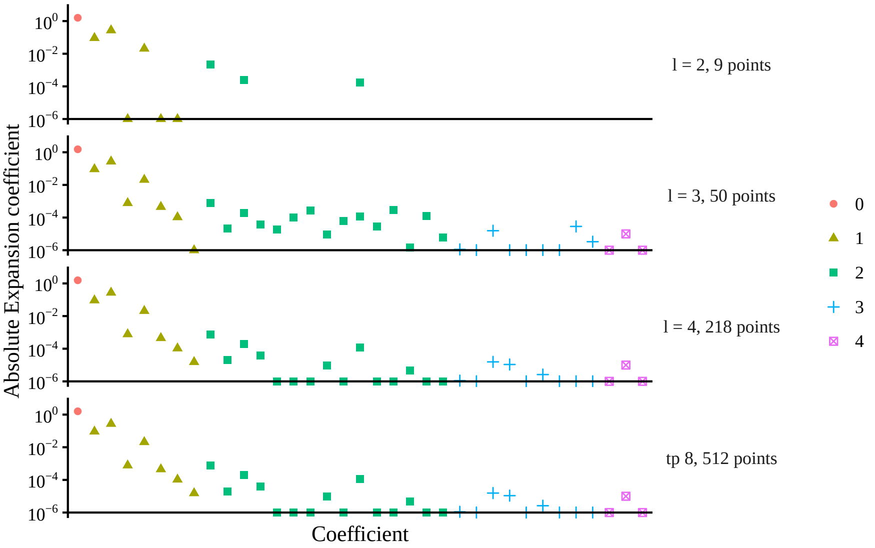


3: Depiction of the points for the $S_3^{(3)}$ quadrature set comprised of the rules in Table 3. The red points are the Q_7 rules in each dimension, and the points and planes in blue are those from the three permutations of the rule $Q_3 \otimes Q_3 \otimes Q_1$. The black points are the two non-redundant points

Black-Scholes w/ Tensor Product Quadrature



Sparse Black-Scholes: Coefficient Estimation



Sparse Quadrature Observations

- We have seen that sparse quadrature can give us accurate expansions with a relatively small number of function evaluations.
- Sparse quadrature does not eliminate the curse of dimensionality, but it does help.
- Could have saved even more evaluations if we had used nested quadrature rules in the Smolyak construction (e.g., trapezoid, Gauss-Kronrod, or Clenshaw-Curtis rules).
  - In the Smolyak construction, the subtractions then remove more points.
- There are further improvements possible as well...

Anisotropic Sparse Quadrature

- In the Black-Scholes example, it appeared that the volatility was more important than the other two input parameters.
- Anisotropic quadratures are a way to handle integrals that require more accuracy in a given dimension.
- A simple way of doing this is to introduce a weight into the selection of quadrature rules.:

$$S^{(d)}_{\ell,\vec{a}} f = \sum_{q=\ell-d}^{\ell-1} (-1)^{\ell-1-q} \binom{d-1}{\ell-1-q} \sum_{q+d-1<\|\vec{k}\|_{\vec{a}}\leq q+d} Q_{2^{k_1}-1} \otimes \cdots \otimes Q_{2^{k_d}-1} f, \tag{25}$$

where $\vec{a}$ is a d-length vector of weights, and $\|\vec{k}\|_{\vec{a}} = \sum_{i=1}^{d} |a_i k_i|$.
- If $\vec{a} = (1, 0.5)$, then the $\ell = 3$ quadrature rule with $d = 2$ would be

$$S^{(d)}_{\ell,(1,0.5)} f = \\ -Q_1 \otimes Q_7 f - Q_1 \otimes Q_{15} f - Q_3 \otimes Q_3 f - Q_3 \otimes Q_1 f + \\ Q_3 \otimes Q_{15} f + Q_3 \otimes Q_7 f + Q_1 \otimes Q_{31} f + Q_7 \otimes Q_3 f + Q_7 \otimes Q_1 f \tag{26}$$

- This rule has a maximum of 31 points in one direction and 7 in the other dimension.

Adaptive Sparse Quadrature

- We could make the quadrature adaptive in each dimension to try and automatically determine which direction to add more points in.
- We would compute a quadrature rule as

$$A^{d})f = \sum_{\vec{k}\in I} \Delta_{2^{k_1}-1} \otimes \cdots \otimes \Delta_{2^{k_d}-1} f, \tag{27}$$

  where I is the set of all indices included in the rule.
- The adaptive algorithm starts with $I = \{(1, \cdots, 1)\}$. Then, we add a point to I with an additional level in the dimension with the largest value of the tensor product of Δ quadratures because the magnitude of a Δ quadrature indicates how much the integral changes when adding new points. Then an additional level in the direction of the level just added. The rule grows by considering those tensor products that are adjacent to terms already in the set.

Section 6

Regression Equations

- The approximation of a quantity using polynomial chaos can be constructed using other means than quadrature.
- One possibility is to think of the expansion as a function to be estimated via regression and use regularized regression techniques to minimize the number of function evaluations needed.
- To set the stage for this approach let us consider an Nth order Hermite expansion of a function $g(x)$,

$$g(x) \approx \sum_{n=0}^{N} c_n He_n(x).$$

Now consider that we have evaluated $g(x)$ at M values of x. The resulting data gives us the following system of equations

$$\begin{aligned}
g(x_1) &= c_0 He_0(x_1) + c_1 He_1(x_1) + \cdots + c_n He_n(x_1) + \epsilon_1, \\
g(x_2) &= c_0 He_0(x_2) + c_1 He_1(x_2) + \cdots + c_n He_n(x_2) + \epsilon_2, \\
&\vdots \\
g(x_M) &= c_0 He_0(x_M) + c_1 He_1(x_M) + \cdots + c_n He_n(x_M) + \epsilon_M.
\end{aligned}$$

Regression Equations

- Here we have written the expansion error for each case as ϵ_i.
- This system is M equations for $N+1$ unknowns, the c_n coefficients, and therefore has no unique solution unless $M = N+1$.
- We can write this system using rectangular matrices as

$$y = Ac,$$

  where y is the vector of length M that contains the $g(x_i)$, A is the $M \times (N+1)$ matrix of the Hermite functions evaluated at x_i, and c is a vector of length $(N+1)$ for the unknown c_i coefficients.
- One could use standard least squares regression to estimate the coefficients by multiplying the equation on both sides by A^t and then solving to get

$$c_{ls} = (A^tA)^{-1}A^ty. \tag{28}$$

- That is, if the inverse of (A^tA) exists, which it will not under many conditions, including when $M < N+1$.
- Using least squares will not necessarily save us the growth in function evaluations as N increases.

Least-Squares as a Minimization Problem

- The least squares solution is the minimizer of the sum of the squares of the residuals, ϵ_i.
- That is, least squares minimizes the function

$$J_{ls}(c) = \frac{1}{2}\sum_{i=1}^{M}(y_i - a_i c)^2, \tag{29}$$

  where a_i is the ith row of A.
- The minimizer of J_{ls} can be found by taking the derivative with respect to c and setting the result to zero, to get

$$\sum_{i=1}^{M} a_i^t (y_i - a_i c) = 0.$$

- This can be re-written in matrix form as,

$$A^t y = A^t A c,$$

  the solution of which is c_{ls}.

Regularization of the Least-Squares Problem

- The idea behind regularized regression is to modify the least-squares problem to make it possible to solve the minimization problem if the inverse of $(A^t A)$ does not exist.
- We will be interested in regularizations where we minimize the magnitude of the coefficients c_i.
- We have seen, that many of the coefficients can be small as in the Black-Scholes example or the coefficients decay to zero as $N \to \infty$.
- A common way to regularize the least squares minimization is to add a penalty term corresponding to the norm of the coefficients

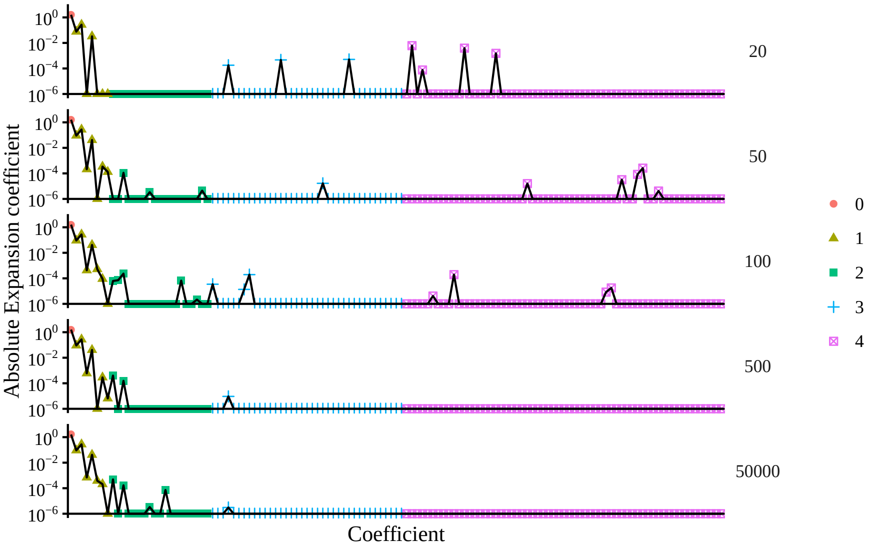
$$J_{el} = \frac{1}{2}\sum_{i=0}^{M}(y_i - a_i c)^2 + \lambda \sum_{n=1}^{N} |c_n| + \lambda \frac{(1-\alpha)}{2} \sum_{n=1}^{N} c_n^2.$$

- The minimizer of this function is called the elastic net minimizer. The parameter $\lambda > 0$ controls the amount of penalty for large coefficients.
- Choosing $\alpha = 1$ only considers the L_1 penalty and is called the lasso penalty, and $\alpha = 0$ is called the ridge penalty.

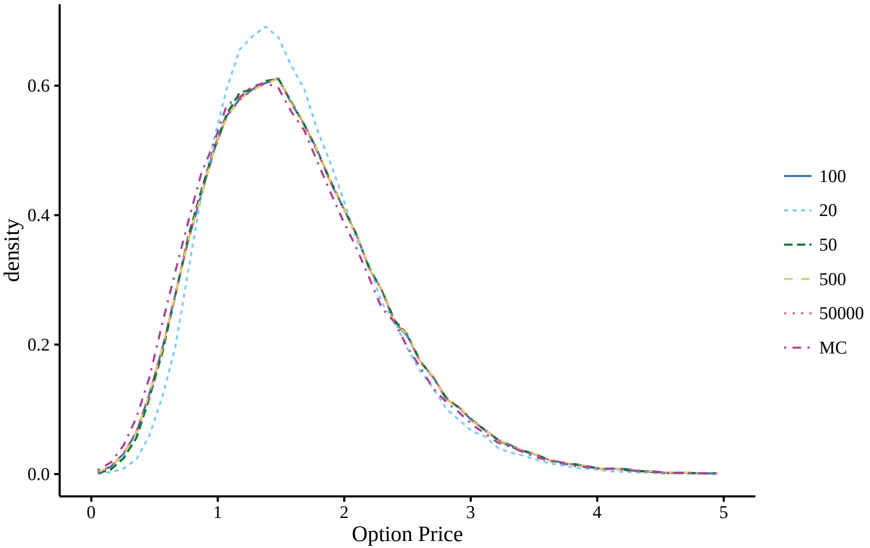
Regularization of the Least-Squares Problem

- The goal of elastic net is to find estimates of the expansion parameters that both approximate the values of $\mathbf{g}(\mathbf{x})$ in the data and constrict the coefficients in the expansion.
- This will work well if many of the correct coefficients are small.
- The balance between correctly matching the data and making the coefficients small is struck through the parameter λ.
- Ideally, we want λ to be as small as possible.
- To obtain λ we typically perform a cross-validation procedure:
  - Use a subset of the data (called training data) to construct an elastic net fit with a given value of λ and then
  - Test the fit on the data *not* used to create the fit (called the testing data).
  - The smallest value of λ that has an acceptable error on the test data is used for the fit.
- One can use the implementation for `R`, called `glmnet`, or using `Scikit-Learn` for `python`. `Matlab` has a function for lasso and ridge penalties.

Black-Scholes w/ Regularized Regression, $\alpha = 0.75$



Black-Scholes w/ Regularized Regression, $\alpha = 0.75$



Conclusion

- In this part we learned how things work in multiple dimensions.
- The curse of dimensionality strikes.
- Can improve the quadrature with sparsity and other tricks.
- Regression can help as well.

Thank you!

Polynomial Chaos Expansions for Uncertainty Quantification

AICES EU Regional School 2016 - Part 2

Ryan G. McClarren

Texas A&M University