8 Forward Uncertainty Quantification

8.1 Motivational Example

Example. Consider a steady-state diffusion problem:

$$\begin{cases} -div(a\,\nabla u) = f \quad \text{on } D \subset \mathbb{R}^d \\ \text{boundary conditions} \end{cases}$$

This PDE can model different thigs: u can be seen as the equilibrium position of a membrane that is loaded by f, or, u can be the temperature in a room, or, u can be the concentration of pollutant in a river.

Let $u = u(x)$ be the solution of the (so-called) elliptic PDE.

Usually, a and f are known fixed coefficients.

However it might be that $a = a(x)$, namely the conductivity field, can be uncertain. Also $f = f(x)$, namely the source, can be uncertain.

This means that we have uncertain inputs that can also be spatial fields (so they become random fields).

Since u depends on a and f, also u is uncertain.

What can be the interest here? What are we interested in evaluating ? u, of course. But also some functional Q of the solution (this is called quantity of interest QoI).

For instance:

$Q(u) = u(\vec{x_0})$ we might be interested in evaluating the solution in a given point x_0

$Q(u) = \frac{1}{|D_0|}\int_{D_0} u(\vec{x})\,d\vec{x}$ we can be interested in evaluate the average of the solution on a given domain or subdomain D_0

(we have a river and $u = u(x)$ is the pollution of the river: we might want to evaluate the average of the concentration in a certain portion of the river)

This is something that, if a and f would not have been uncertain, would just involved finite element, .. (numerical scheme computation)

However, the problem is that a and f are uncertain. This is the new assumption.

We first need to describe uncertainty (in Chapter 3 we discussed about how to parametrized random inputs).

Thus, if we consider a probabilistic approach we need to deal with **random inputs**.

This means that each new realization of these inputs lead to new solution. Therefore, we might be interested in evaluate some statistics of these QoI (expectation, confidence intervals, secondo moment, ..).

We have two cases (for a and f being uncertain):

1. coefficients are spatially dependent $\Rightarrow$ **random fields**
2. coefficients are not spatially dependent $\Rightarrow$ **random input parameters**

We also know that by using Karhunen-Loeve expansion (K-L expansion) (chapter 3), we can always turn the description of random fields into a set of parameters. (Parametrization of random fields). In conclusion we can somehow always end up in the case of random input parameters.

For example, we could consider $\log(a)$) as a random field, for example a Gaussian random field. Then we can parametrize it with the truncated K-L expansion.

Se qualcuno è interessato, Other examples are present in section 8.1 (3 examples).
There are some challenges that arises.

1. If a is a random field then a depends on a spatial cordinate x and on some random quantity that can be parametrized with ω, therefore: $a = a(x, \omega)$. a might have low regularity (e.g. if a represents the property of a certain material then the material might not be homogeneous in every point, the characteristic a might be around a certain mean but with little differences in different zones). If a is unregular, the finite-element mesh that is required should be very fine (h =mesh size should be very small), but this implies a very large linear system (that we transform the PDE problem into) to be solved $\Rightarrow$ complexity.
2. The random field can be characterized by large variance. This means that the contrast, namely the ratio between $a_{\max}$ and $a_{\min}$ can be very large ($\frac{a_{\max}}{a_{\min}}$ very large), which is related with the continuity and coercitivity constants. These latter constants appear in error estiamtes and having those very large means that we may have unstable solutions.
3. The correlation length of the field, say λ, may be very small (if properties are very changing in space). In chapter 3 we saw that the decay of the of the singular values of the K-L expansion of a field was strictly related with the fact that covariance function has large/small correlation length. The smaller the correlation length (the less correlated are two values in space for two points in space that are close to each other) -¿ highest number of K-L nodes that we need to consider to properly approximate (parametrize) the random field -¿ if this number is high then the stochastic dimension (number of input parameters), say p, will be very large.

Maybe not all of these 3 challenges appear at the same time, but some of them might.

8.2 A *more informal* way of using Monte Carlo techniques

What we learned about MC in Chapter 4?

Consider a generic problem $\mathcal{P}_h$.

There are some random inputs $\vec{Y}(\omega) \in \mathbb{R}^p$, where p is dimension of the parameter space and where ω underlies the fact that the inputs are random.

While $\vec{Y}(\omega)$ is a random vector, $\vec{y}$ is a realization of $\vec{Y}(\omega)$.

Using a discretization technique (for instance the finite element), to each realization of the random input we get a solution $U_h(\omega) \in \mathbb{R}^{N_h}$ (since we have ω we see that this depends on the random realizations). This is the state vector and it has been obtained by solving a discretized problem (finite elements, finite differences, finite volume, ..).

On the state vector we might be interested on computing a QoI (an output) and this will be a scalar (for the sake of convenience): $Q_{h,p}(\omega) \in \mathbb{R}$. This depends on both h =discretization paramenter, and p =dimension of the random input.

??????

The QoI (for which we may want to compute expectations, etc) depends on the solution of a complex problem. Here we clearly see why we need a simplification on models.

????????????

The objects that we are dealing with are:

$\vec{Y}$: random input, (it can be a multivariate gaussian random vector)

$\vec{U}_h$: numerical solution of a PDE, or an ODE, ..

here we use capital letter because it is a random quantity

$Q_{h,p}$: a (non) linear function of $\vec{U}_h$

The **true** QoI $Q(\omega)$ is not accessible because the exact solution of the problem is not accessible (we do not know the exact solution of the problem, namely $\vec{U}$, we only know $\vec{U}_h$).

But, what we can assume is that:

- $\mathbb{E}[Q_{h,p}] \to [\substack{h\to 0 \\ (p\to\infty)}]\mathbb{E}[Q]$

($p \to \infty$ if we have truncated the K-L expansion of a field)

Notice that we have now two sources of error: an error related with the discretization and there is a statistical error why? because the expectations will be approximated by MC estimates.

- $|\mathbb{E}[Q_{h,p} - Q]| = O(h^{\alpha}) + O(p^{-\alpha'})$

for certain values of α and α'.

(In the case of a finite number of parameters: $|\mathbb{E}[Q_{h,p} - Q]| = O(h^{\alpha})$)

The standatd (crude) Monte Carlo estimator of $\mathbb{E}[Q]$ will be:

$\hat{Q}_{CMC} = \frac{1}{N}\sum_{i=1}^{N} Q_{h,p}^{(i)}$

(the point is that we can only compute the discrete QoI (QoI that depends on discrete solution))

where $\{Q_{h,p}^{(i)}\}_{i=1}^{N}$ are iid samples computed by solving the numerical problem $\mathcal{P}_h$.

Spoiler: the convergence of the crude MC technique will tell us:

$$MSE := \mathbb{E}[(\hat{Q}_{CMC} - \mathbb{E}[Q])^2] = Var(\hat{Q}_{CMC}) + (\mathbb{E}[\hat{Q}_{CMC}] - \mathbb{E}[Q])^2 = \underbrace{\frac{Var(Q_{h,p})}{N}}_{\text{sampling error}} + \underbrace{(\mathbb{E}[\hat{Q}_{CMC}] - \mathbb{E}[Q])^2}_{\text{model error (discretization)}}$$

The total error is split in: sampling error and model error (due to discretization).

The task is to understand how to balance the two sources of error.

A typical case is when $\alpha = 1$, therefore:

$MSE = O(\frac{1}{N}) + O(h^2)$

(sampling error + model error)

But if we want $MSE \leq \varepsilon^2$ then we have to take $h \sim \varepsilon$, $N \sim \varepsilon^{-2}$.

This means that if $\varepsilon = 10^{-3}$, for example, then $h \sim 10^{-3}$ and $N \sim 10^6$.

In this setting the MC cost $= O(N h^{-d}) = O(\varepsilon^{-(d+2)})$

If $d = 2$ = spatial dimension, then $O(10^{12})$.

Everything quickly becomes proibitively expensive. So, we need to introduce Multilevel Monte Carlo.

8.3 A "simple" ODE example

Consider the **Lodka-Volterra (predator-prey) model**.

$$\begin{cases} \dot{\vec{u}} = \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \end{bmatrix} = \begin{bmatrix} \theta_1 u_1 - \theta_{12} u_1 u_2 \\ \theta_{21} u_1 u_2 - \theta_2 u_2 \end{bmatrix} = \vec{f}(\vec{u}) \qquad t \in (0, T) \\ \vec{u}(0) = \vec{u}_0 \end{cases}$$

This is a Cauchy problem (initial value problem) for a non linear syste of 2 (coupled) ODEs.

Where:

$u_1 = \#$ prays

$u_2 = \#$ predators

$\theta_1, \theta_2, \theta_{12}, \theta_{21} > 0$

Assume (for the sake of simplicity) that:

$\theta_1 = \theta_2 = \theta_{12} = \theta_{21} = 1$

and that only $\vec{u}_0$ (initial condition) is uncertain.

For example we can take $\vec{u}_0 \sim \mathcal{U}(\Gamma)$ where $\Gamma = \overline{\vec{u}_0} + [-\delta, \delta]^2$ (i.e. a square centered at $\overline{\vec{u}_0}$).

The goal is to estimate the average value of the number of prays at time T:

$\mathbb{E}[u_1(T)]$ at time $T > 0$

We do not know the exact solution $\vec{u} = \vec{u}(t)$. So, we rely on a numerical solver, e.g. explicit euler.

This means that after j time steps of length $h = \frac{T}{N_h}$ (drawing)

we have:

$\vec{u}_j = \vec{u}_{j-1} + h\,\vec{f}(\vec{u}_{j-1}), \quad j = 1, .., N_h$

starting from $\vec{u}_0 = \vec{u}_0(\omega)$ (where ω underlies that $\vec{u}_0$ is a random input).

The explicit euler scheme is consistent of order 1 and convergent. This means that:

$\exists\, K > 0$: $\quad ||\vec{u}(T) - \vec{u}_{N_h}|| \leq K\,h$

(where $t_{N_h} = T$)

Our QoI is:

$Q = \phi(\vec{u}(T)) = u_1(T)$ (first component of $\vec{u} = [u_1, u_2]^T$)

which means that:

$Q_h = \phi(\vec{u}_{N_h})$.

(evaluation of ϕ on the numerical object).

We want to estimate $\mathbb{E}[Q_h]$ using the Monte Carlo method.

It's very simple here: we pick different random realization of $\vec{u}_0$ (initial condition), we solve the problem (using for instance explicit euler) and the pick the first component of the solution at time T. Then we do the mean.

Remark: the QoI ϕ is Lipschitz-continuous with constant $L = 1$.

This allows us to say that:

$|\mathbb{E}[Q] - \mathbb{E}[Q_h]| = |\mathbb{E}[Q - Q_h]| \leq \mathbb{E}[||\vec{u}(T) - \vec{u}_{N_h}||] \leq K\,h$

since $Q - Q_h \leq \vec{u}(T) - \vec{u}_{N_h}$ (because of $L = 1$) (??????????????????)

the last inequality is due to the fact that explicit euler shceme is consistent of order 1 (as seen before).

Now Monte Carlo comes into play!

$\hat{Q}_h = \hat{Q}_{h,N} := \frac{1}{N}\sum_{i=1}^{N} Q_h^{(i)}$

$()\hat{Q}$ = Monte Carlo estimator)

with N iid samples $Q_h^{(1)}, .., Q_h^{(N)}$ of Q_h, meaning that we have to solve the ODE system N times.

With small h and large N we expect better approximations.

(small h bounds $|\mathbb{E}[Q - Q_h]]|$ and large N approximates better Monte Carlo)

As we already saw, there is a combination of discretization errors + statistical (sampling) errors.

To balance the error contributions we recall the **Bias-Variance decomposition**.

The mean square error (MSE) can be expanded as:

$\mathbb{E}[(\mathbb{E}[Q] - \hat{Q}_h)^2] = (\mathbb{E}[Q - Q_h])^2 + \frac{Var(Q_n)}{N}$.

($\hat{Q}_h$ = Monte Carlo estimator)

proof.

Observe that Monte Carlo is unbiased and so $\mathbb{E}[\hat{Q}_h] = \mathbb{E}[Q_h]$.

Moreover, we recall that $\mathbb{E}[Q]$ is not accessible because we don't know the exact solution.

Then:

$$\begin{aligned}
\mathbb{E}[(\hat{Q}_h - \mathbb{E}[Q])^2] &= \mathbb{E}[(\hat{Q}_h \pm \mathbb{E}[Q_h] - \mathbb{E}[Q])^2] \\
&= \mathbb{E}[(\hat{Q}_h - \mathbb{E}[Q_h])^2] + \mathbb{E}[(\mathbb{E}[Q_n] - \mathbb{E}[Q])^2] + \underbrace{2\,\mathbb{E}[(\hat{Q}_h - \mathbb{E}[Q_h])(\mathbb{E}[Q_h] - \mathbb{E}[Q])]}_{=0} \\
&= \underbrace{\mathbb{E}[(\hat{Q}_h - \mathbb{E}[\hat{Q}_h])^2]}_{\substack{\text{we exploited:} \\ \mathbb{E}[\hat{Q}_h] = \mathbb{E}[Q_h]}} + (\mathbb{E}[Q_h - Q])^2 \\
&= \underbrace{Var(\hat{Q}_h)}_{\substack{\text{we exploited that} \\ \hat{Q}_h \text{ is unbiased}}} + (\mathbb{E}[Q_h - Q])^2 \\
&= \frac{Var(Q_h)}{N} + (\mathbb{E}[Q_h - Q])2
\end{aligned}$$

■

Back to our case (ODE example), we assume that $Var(Q_h) \leq \sigma^2$ (independently of h) and, recalling that $|\mathbb{E}[Q] - \mathbb{E}[Q_h]| = |\mathbb{E}[Q - Q_h]| \leq k\,h$, we get:

$MSE = \mathbb{E}[(\mathbb{E}[Q] - \hat{Q}_h)^2] = (\mathbb{E}[Q_h - Q])^2 + \frac{Var(Q_h)}{N} \leq k^2\,h^2 + \frac{\sigma^2}{N}$

This is a first bound that we can get.

A second bound is:

$error_{h,N} = e_{h,N} := |\mathbb{E}[Q] - \hat{Q}_h| \leq \underbrace{|\mathbb{E}[Q] - \mathbb{E}[Q_h]|}_{\text{discretization}} + \underbrace{|\mathbb{E}[Q_h] - \hat{Q}_h|}_{\text{MC error}}$

At which price? (computationally speaking, we are able to perform MC analysis)? **Monte Carlo complexity analysis for the prey-predator problem.**

The cost of each time step is 8 flops:

$Cost(\hat{Q}_h) = 8\,N_h \cdot N = 8\,\frac{T}{h}N$ flops

N times we evaluate N_h steps (since we want the time T for which we have $t_{N_t} = T$) times 8, because each time step is 8 flops.

The total cost to compute a standard MC estimator for $\mathbb{E}[u_1(T)]$ with the explicit Euler discretization (in the prey-predator model) such that $MSE < \varepsilon^2$ satisfies:

$Cost(\hat{Q}_h) = O(\varepsilon^{-3})$.

Namely, the cost scales as ε to the power -3.

Why?

We recall

$MSE \leq k^2 h^2 + \frac{\sigma^2}{N}$

Therefore:

$MSE \leq k^2 h^2 + \frac{\sigma^2}{N} < \varepsilon^2$

assuming that $k^2 h^2 < \frac{\varepsilon^2}{2}$, $\frac{\sigma^2}{N} \frac{\varepsilon^2}{2}$:

$$\Rightarrow \begin{cases} k\,h \sim \frac{\varepsilon}{\sqrt{2}} \\ \frac{\sigma}{\sqrt{N}} \sim \frac{\varepsilon}{\sqrt{2}} \end{cases} \Rightarrow \begin{cases} h \sim \frac{\sqrt{2}}{k}\varepsilon \\ N \sim 2\,\sigma^2\,\frac{1}{\varepsilon^2} \end{cases}$$

$$\Rightarrow Cost(\hat{Q}_h) = 8\,\frac{T}{h}N \sim 8\,\frac{T}{\frac{\sqrt{2}}{k}\varepsilon}(2\,\sigma^2\,\frac{1}{\varepsilon^2}) \sim (k\,\frac{16}{\sqrt{2}}\,\sigma^2\,T)\,\varepsilon^{-3}$$

We see that the longer the time interval (the bigger T), the higher the cost.

We can bound the error, we can highlight two error components, we can perform a MC cost analysis highlighting the dependence on the accuracy leve (the treshold).

8.4 Monte Carlo method rivisited

Discretization errors and sampling errors combined together

Objects we will deal with:

$\vec{y}$: vector of random inputs

$u(\vec{y})$: solution of the problem

$Q(\vec{y}) = Q(u(\vec{y}))$: quantity of interest

The goal is to approximate $\mathbb{E}[Q(u(\vec{y}))]$.

In principle, Monte Carlo would tell us:

$\mathbb{E}[Q(\vec{y})] \approx \frac{1}{N}\sum_{i=1}^{N} Q(u(\vec{y}^{(i)}))$

so the QoI is evaluated on N solutions $u(\vec{y}^{(i)})$, $i = 1, .., N$ of the problem.

Since $u = u(\vec{y})$ is not known (not *accessible*), we need to rely on a discretization technique.

By discretizing the problem:

$\mathbb{E}[Q(u(\vec{y}))] \approx \hat{Q}_h = \frac{1}{N}\sum_{i=1}^{N} Q(u_h(\vec{y}^{(i)}))$

and the errors can be split as:

$$\begin{aligned} \mathbb{E}[Q(u(\vec{y}))] - \hat{Q}_h &= \mathbb{E}[Q(u(\vec{y})) \pm Q(u_h(\vec{y}))] - \hat{Q}_h \\ &= \mathbb{E}[Q(u(\vec{y})) \pm Q(u_h(\vec{y}))] - \tfrac{1}{N}\textstyle\sum_{i=1}^{N} Q(u_h(\vec{y}^{(i)})) \\ &= \underbrace{\mathbb{E}[Q(u(\vec{y})) - Q(u_h(\vec{y}))]}_{\varepsilon^Q(h)} + \underbrace{\mathbb{E}[Q(u_h(\vec{y}))] - \tfrac{1}{N}\textstyle\sum_{i=1}^{N} Q(u_h(\vec{y}^{(i)}))}_{\varepsilon_h^Q(N)} \end{aligned}$$

where:

$\varepsilon^Q(h)$ = discretization error (or bias)

$\varepsilon_h^Q(N)$ = statistical error (or MC error, or sampling error).

Note that $\hat{Q}_h$ is a biased estimator. In fact:

$\mathbb{E}[\hat{Q}_h] = \mathbb{E}[Q(u_h(\vec{y}))] \neq \mathbb{E}[Q(u(\vec{y}))]$.

(because of the discretization error).

The goal is to estimate the two error contributions separately.

Lec 2. ———————————

How information of the problem and information on discretization technique can help in estimating the discretization error? How can these guys interplay with the statistical error?

Let's start with the **discretization error**.

What are the ingredients that we are going to exploit?

1. We need some knowledge about an error estimate on the solution (e.g. previously we used the explicit Euler scheme and we could conclude that $|..| < k\,h$)

   whatever the method we use, we need to rely on some knowledge from the approximation process that we use

   namely:

   very generically:

   $||u(\vec{y}) - u_h(\vec{y})||_V \leq c_u\,h^\alpha$

   E.g. in explicit Euler $\alpha = 1$

2. We assume that the functional Q is regular, in particular we assume that Q is Lipschitz-continuous:

   $|Q(u) - Q(v)| \leq c_Q\,||u - v||_V \quad \forall u, v \in V$, where V is the (proper) functional space in which we are framing the problem

   Furthermore, we assume that $Q(0) = 0$.

By definition, the discretization error is:

$$\begin{aligned} |\varepsilon^Q(h)| &= \left|\,\mathbb{E}[Q(u(\vec{y})) - Q(u_h(\vec{y}))]\,\right| \\ &= \left|\,\textstyle\int (Q(u(\vec{y})) - Q(u_h(\vec{y})))\,f_y(\vec{y})\,d\vec{y}\,\right| \quad f_y(\cdot) \text{ is the pdf of the random input vector } \vec{y} \\ &\leq c_Q\,\textstyle\int ||u(\vec{y}) - u_h(\vec{y})||_V\,f_y(\vec{y})\,d\vec{y} \quad \text{because of 2.} \\ &\leq c_Q\,c_u\,h^\alpha \quad \text{because of 1.} \end{aligned}$$

Warning: c_u might depend on $\vec{y}$ through some stronger norm $||u(\vec{y})||$

Let's consider now the **statistical error**:

$$\begin{aligned} \varepsilon_h^Q(N) &= \mathbb{E}[Q(u_h(\vec{y}))] - \tfrac{1}{N}\textstyle\sum_{i=1}^{N} Q(u_h(\vec{y}^{(i)})) \\ &= \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \left(\mathbb{E}[Q(u_h(\vec{y}))] - Q(u_h(\vec{y}^{(i)}))\right) \end{aligned}$$

Notice that here we have $u_h(\vec{y})$ and so, we no longer have the gap between $u(\vec{y})$ and $u_h(\vec{y})$. This means that:

$\mathbb{E}[\varepsilon_h^Q(N)] = 0$

$Var(\varepsilon_h^Q(N)) = \frac{1}{N} Var(Q(u_h(\vec{y})))$

with N iid samples.

We need to show that $Var(\varepsilon_h^Q(N))$ is finite to be able to use the central limit theorem.

So, to conclude that the MC error is bounded by (the result from the CTL):

$|\varepsilon_h^Q(N)| \leq z_{1-\frac{\alpha}{2}} \frac{\sqrt{Var(Q(u_h(\vec{y})))}}{\sqrt{N}}$

we need to prove that $Var(Q(u_h(\vec{y})))$ is finite (CTL hypothesis).

Let us observe that:

$Var(Q) = \mathbb{E}[(Q - \mathbb{E}[Q])^2] = \mathbb{E}[Q^2] - (\mathbb{E}[Q])^2 \leq \mathbb{E}[Q^2]$

that is (in terms of the problem that we are studying):

$Var(Q(u_h(\vec{y}))) \leq \mathbb{E}[Q(u_h(\vec{y}))^2]$.

But we also know that:

$||u(\vec{y}) - u_h(\vec{y})||_V \leq c_u(\vec{y})\, h^\alpha$

for a certain value of $\alpha > 1$.

Here we are more precise and we write explicitly the dependence on $\vec{y}$.

We take the sqaure and we get:

$||u(\vec{y}) - u_h(\vec{y})||_V^2 \leq c_u^2(\vec{y})\, h^{2\alpha}$

we now take the mean:

$\mathbb{E}[||u(\vec{y}) - u_h(\vec{y})||_V^2] \leq \mathbb{E}[c_u^2(\vec{y})]\, h^{2\alpha}$

and so, considering that Q is Lipschitz and that $Q(0) = 0$ we get:

$Q(u_h(\vec{y})) - Q(0) \leq c_\alpha ||u_h(\vec{y}) - 0||_V$

$\Rightarrow\ \mathbb{E}[Q(u_h(\vec{y}))^2] \leq c_\alpha^2\, \mathbb{E}[||u_h(\vec{y})||_V^2]$

Therefore:

$$\begin{aligned} Var(Q(u_h(\vec{y}))) &\leq \mathbb{E}[Q(u_h(\vec{y}))^2] \\ &\leq c_Q^2\, \mathbb{E}[||u_h(\vec{y})||_V^2] \\ &= c_Q^2\, \mathbb{E}[||u_h(\vec{y}) \pm u(\vec{y})||_V^2] \\ &\leq 2\, c_Q^2 \left(\mathbb{E}[||u(\vec{y})||_V^2] + \mathbb{E}[||u(\vec{y}) - u_h(\vec{y})||_V^2]\right) \\ &\leq 2\, c_Q^2 \Big(\underbrace{\mathbb{E}[||u(\vec{y})||_V^2]}_{<\infty\ (*)} + \mathbb{E}[c_u^2(\vec{y})]\, h^{2\alpha} \Big) < \infty \end{aligned}$$

$(*)$ is $< \infty$ because we know that the problem can be solved.

The 2 is gained because of the triangular inequality and ^{2}. (???????????)

The variance is finite $\Rightarrow$ we can apply CLT $\Rightarrow$ we have a bound on the MC error.

In conclusion: for Lipschitz functionals Q we can bound the statistical error as we did in Chapter 4: asymptotically, with probability $1 - \alpha$:

$|\varepsilon_h^Q(N)| = |\mathbb{E}[Q(u_h(\vec{y}))] - \frac{1}{N}\sum_{i=1}^N Q(u_h(\vec{y}^{(i)}))| \leq z_{1-\frac{\alpha}{2}} \frac{\sqrt{Var(Q(u_h(\vec{y})))}}{\sqrt{N}}$.

Remark: Regarding the mean squared error (*check a previous result*):

$MSE(\hat{Q}_h) = \mathbb{E}[(\hat{Q}_h - \mathbb{E}[Q(u_h(\vec{y}))])^2] = \underbrace{\left(\mathbb{E}[Q(u(\vec{y})) - Q(u_h(\vec{y}))]\right)^2}_{(\text{bias})^2} + \underbrace{Var(\hat{Q}_h)}_{\frac{Var(Q(u_h(\vec{y})))}{N}}$

Now we have all the tools to perform a **Monte Carlo complexity analysis**.

Notes. 8.3.3.

Chapter 8

Forward Uncertainty Quantification

Mathematical models and computer simulations are widely used in engineering and science applications. However, in many cases, the parameters in the model are affected by uncertainty, either because they are not perfectly known or because they are intrinsically variable. The goal of Uncertainty Quantification (UQ) tools is to devise effective ways to include and treat uncertainty in a mathematical model. Relevant examples are, for instance: modeling of living tissues / biological fluids, combustion problems, weather forecast and climate modeling, subsurface modeling (e.g. groundwater flows, contaminant transport, earthquake simulations, and so on).

We have seen so far how the simulation of statistical models provides an effective tool to describe and propagate uncertainty. We will now touch the case of mathematical models based on Partial Differential Equations (PDEs) whose input data (coefficients, forcing terms, boundary conditions, domain boundary, ...) are uncertain and described as random variables or random fields; similar consideration hold for dynamical systems described by Ordinary Differential Equations (ODEs). Throughout this section, the term *inputs* will be used to denote parameters, initial conditions, boundary conditions, source terms that exhibit uncertainties which must be determined and propagated through models to construct predictions with quantified uncertainties. Therefore,

> We will look at the solution of a mathematical model (such as a PDE) as to a random function, $u = u(\omega, x)$; denoting by ω an elementary random event, to highlight the dependence on *generic* random inputs. For the sake of computations, we will always *parametrize* the random inputs in terms of a random vector $\mathbf{y}$ (see Chapter 3).
>
> One of the main questions we will address is how to effectively approximate some (random) output quantities of interest $Q(u)$, or the random function $u(\omega, x)$ itself, and how to propagate uncertainty from the inputs to the output quantities of interest.

8.1 Three Examples

Before addressing the problem in an abstract form, let us consider three possible examples of interest.

Example 1: linear elasticity with random elastic properties

The linear elastic deformations of an isotropic solid occupying the domain $D \subset \mathbb{R}^d$ is described in terms of the stress tensor $\boldsymbol{\sigma} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$, the strain tensor $\boldsymbol{\varepsilon} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$, the body force $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d$ and the displacement field $\boldsymbol{u} : \mathbb{R}^d \to \mathbb{R}^d$, this latter being the unknown of the problem.

The governing equations consist of an equation stating the equilibrium of forces

$$-\text{div}(\boldsymbol{\sigma}) = \boldsymbol{f} \quad \text{in } D,$$

the strain-displacement relation

$$\boldsymbol{\varepsilon}(\boldsymbol{u}) = \frac{1}{2}(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)$$

and the constitutive law, which in the linear isotropic case takes the form

$$\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon}(\boldsymbol{u}) + \lambda(\text{div}(\boldsymbol{u}))\boldsymbol{I}.$$

Here μ and λ are the Lamé coefficients, which can be expressed in terms of the Young modulus E and the Poisson coefficient ν as

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}.$$

The equilibrium problem for a linear elastic material can therefore be written as follows:

$$\begin{cases} -\text{div}(\mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T) + \lambda(\text{div}\boldsymbol{u})\boldsymbol{I}) = \boldsymbol{f} & \text{in } D \\ \boldsymbol{u} = \boldsymbol{g} & \text{on } \Sigma_D \\ \boldsymbol{\sigma}\boldsymbol{n} = \boldsymbol{h} & \text{on } \Sigma_N. \end{cases} \tag{8.1}$$

On the Dirichlet boundary Σ_D we impose a prescribed displacement, whereas on the Neumann boundary Σ_N we impose that the normal stress $\boldsymbol{\sigma}\boldsymbol{n}$ equals a prescribed traction vector $\boldsymbol{h}$.

We may assume that the Young modulus E, the Poisson ration ν and the load vector $\mathbf{f} = (f_1, f_2, f_3)$ are uncertain and treated as random variables, through a vector of random parameters: $\mathbf{y} = (E, \nu, f_1, f_2, f_3)$, with $E > 0$ and $0 < \nu < 1/2$. Hence, we can prescribe the following *uncertainty model*: $\mathbf{y}$ takes values in $\Gamma \subset \mathbb{R}_+ \times (0, 1/2) \times \mathbb{R}^3$ and has joint probability density function $f_{\mathbf{y}}(\mathbf{y}) : \Gamma \to \mathbb{R}_+$.

In this case, for any outcome of the random vector $\mathbf{y} \in \Gamma$ there exists a unique solution $\boldsymbol{u} = \boldsymbol{u}(\mathbf{y}) \in V = [H^1_{\Sigma_D}(D)]^3$. We can therefore define the random map $\boldsymbol{u} : \Gamma \to V$ that depends on $P = 5$ random parameters.

Example 2: groundwater flow in random porous media

According to Darcy's law, the pressure gradient ∇p and the fluid velocity $\boldsymbol{u}$ in a porous medium follow a linear relation, that is

$$\begin{cases} \boldsymbol{u} = -k\nabla p & \text{in } D \\ \text{div}\boldsymbol{u} = f & \text{in } D \\ +b.c. & \text{on } \partial D \end{cases}$$

The second equation (mass conservation) relates sinks and sources of flow to the velocity field. In most aquifers, the macroscopic properties (porosity and permeability) of the ground are highly variable and never perfectly known. They can be described as random fields, i.e. the permeability $k(x_i) > 0$ at each point $x_i \in D$ is a random variable and, taken n points $x_1, \dots, x_n$, the random variables $k(x_1), \dots, k(x_n)$ are in general correlated.

To ensure tat the permeability field is positive, we can define it as $k = e^Y$, where the *log-permeability* is given by

$$Y(x) = \log(k(x)).$$

This latter is a *random field*, that is a collection of infinitely many random variables. The *uncertainty model* is given in this case by the probability law of the random field, however random

fields must be treated in practice in a suitable way. Indeed, we know that any random field on a compact domain $D \subset \mathbb{R}^d$ with bounded variance and continuous covariance can be expanded in Karhunen-Loève series as

$$Y(\omega, x) = \mathbb{E}[Y](x) + \sum_{i=1}^{\infty} y_i(\omega) b_i(x)$$

where $\mathbf{y} = (y_1, y_2, \ldots)$ is an infinite (countable) sequence of (uncorrelated) random variables.

Hence, for each outcome of the random sequence $\mathbf{y}$, and hence each realization of the random field $\mathcal{Y}(x)$, the PDE problem admits a unique solution $(p(\mathbf{y}), \boldsymbol{u}(\mathbf{y})) \in V = H^1(D) \times H_{div}(D)$.

We can therefore define the random map $(p(\mathbf{y}), \boldsymbol{u}(\mathbf{y})) : \Gamma \subset \mathbb{R}^{\mathbb{N}} \to V$ that depends on an infinite countable number of random variables.

Example 3: a diffusion problem with random piecewise constant coefficients

A simplification of the problem addressed in the previous case is the following diffusion problem:

$$\begin{cases} -(k(x, \mathbf{y})u'(x, \mathbf{y}) = f(x, \mathbf{y}) & x \in D = (a, b) \\ u(a, \mathbf{y}) = u(b, \mathbf{y}) = 0 \end{cases}$$

where the diffusion coefficient and the source term are given by

$$f(x, \mathbf{y}) = 1 + y_1 \cdot \mathbf{1}_{\{x<1/2\}}(x) + y_2 \cdot \mathbf{1}_{\{x \geq 1/2\}}(x), \qquad f(x, \mathbf{y}) = 1 + y_3,$$

respectively, with $y_1, y_2, y_3 \sim \mathcal{U}(-1, 1)$. We know that this problem admits a unique solution in $V = H_0^1(a, b)$ for each outcome of the random vector $\mathbf{y} \in \mathbb{R}^3$; we can therefore define the random map $\boldsymbol{u}(\mathbf{y}) : \Gamma \subset \mathbb{R}^3 \to V$ that depends on $P = 3$ random variables.

8.2 An abstract framework

Solutions of differential equations with uncertain parameters are random processes. All these inputs can be related to physical coefficients, source terms, boundary/initial conditions, or even to the physical domain $D \subset \mathbb{R}^d$, $d = 1, 2, 3$, wherein the PDE has to be solved.

In turn, these quantities can be modeled through *random* input parameters. It is clear that, from a practical standpoint, we need to deal with a vector of P random input parameters, even if the random input is related to a spatially-distributed field, by assuming that we can *parametrize* random fields by a finite number of random variables.

For instance, consider the following model problem, derived from Example 2 (Darcy problem): a scalar, linear, elliptic PDE depending on a random input,

$$\begin{cases} -\text{div}(k(\mathbf{x}; \omega)\nabla u(\mathbf{x})) = f(\mathbf{x}) & \text{in } D \\ u(\mathbf{x}) = h(\mathbf{x}) & \text{on } \partial D, \end{cases} \tag{8.2}$$

where $k = k(\mathbf{x}; \omega)$ is a (positive) random field, that is, *(i)* for a fixed point $\mathbf{x} \in D$, $k(\mathbf{x}; \cdot)$ is a random variable over the outcome space[1] Ω; *(ii)* for a fixed $\omega \in \Omega$, $k(\cdot; \omega)$ is a realization of the random field in D. A common assumption, as in Example 2, is that the random field $k(\mathbf{x}; \omega)$ can be expressed as an infinite sum of uncorrelated random variables $\{\psi_j(\omega)\}_{j=1}^{\infty}$, for a given set of real-valued functions $\{k_j(\mathbf{x})\}_{j=1}^{\infty}$. Since only few modes are *usually* relevant, a low-dimensional representation of a random field is obtained by truncating the infinite sum and retaining a (hopefully small) finite number P of terms:

$$k_P(\mathbf{x}; \omega) = \sum_{j=1}^{P} y_j(\omega) k_j(\mathbf{x}),$$

[1] Here $(\Omega, \mathcal{A}, P)$ is a complete probability space; in particular, Ω is the set of outcomes, $\mathcal{A}$ is the sigma-algebra of subsets of Ω, $P : \mathcal{A} \to [0, 1]$ is a probability measure. Note that when dealing with distributed random fields, P could go to infinity! However, we will avoid to be too technical regarding the probabilistic setting.

provided the variables $y_1, y_2, \ldots$ are uncorrelated. These variables can thus play the role of *random input parameters* $\mathbf{y} = (y_1, \ldots, y_P)$.

Hence, in our *abstract* framework, we consider a PDE problem under the form:

$$\text{find } u \; : \quad \mathcal{L}(\mathbf{y}(\omega))(u(\omega)) = \mathcal{F}(\mathbf{y}(\omega)) \qquad \text{in } D \subset \mathbb{R}^d$$

with suitable boundary/initial conditions. The data of the problem (operator $\mathcal{L}$, forcing term $\mathcal{F}$ and, eventually, the domain D, the initial and boundary conditions) depend on a vector of P random variables:

$$\mathbf{y}(\omega) = (y_1(\omega), \ldots, y_P(\omega)) : \Omega \to \mathbb{R}^P;$$

then, the solution to the PDE problem is also random.

Moreover, u depends on the random event ω only through the random vector $\mathbf{y}(\omega)$, i.e., $u = u(\mathbf{y}(\omega))$. Let us denote by:

- $D \subset \mathbb{R}^d$ is the physical domain where we set our problem, i.e. $\mathbf{x} \in D$;
- $\Gamma = \mathbf{y}(\Omega) \subset \mathbb{R}^P$ the image of $\mathbf{y}$;
- For an integrable function $\Psi : \Gamma \to \mathbb{R}$,

$$\mathbb{E}\left[\Psi(\mathbf{y})\right] = \int_\Gamma \Psi(\mathbf{y}) f_y(\mathbf{y}) d\mathbf{y}.$$

- Similarly,

$$\|w\|_{L^p_{f_y}} = \left(\int_\Gamma w^p(\mathbf{y}) f_y(\mathbf{y}) d\mathbf{y}\right)^{1/p} = \mathbb{E}\left[w^p\right]^{1/p}$$

and

$$L^p_\rho(\Gamma) = \{w : \Gamma \to \mathbb{R} \; : \|w\|_{L^p_{f_y}} < \infty\}.$$

- for any Hilbert space V of real valued functions from D,

$$\|v\|_{L^p_{f_y}(\Gamma;V)} = \left(\int_\Gamma \|v(\mathbf{y})\|^p_V(\mathbf{y}) f_y(\mathbf{y}) d\mathbf{y}\right)^{1/p}$$

and (Bochner space)

$$L^p_\rho(\Gamma; V) = \{v : \Gamma \to V \; : \|v\|_{L^p_\rho(\Gamma;V)} < \infty\}.$$

This means that we can look at the solution of the PDE as a function $u : \Gamma \to V$ taking values in a Hilbert space.

Finally, we will assume that for any $\mathbf{y} \in \Gamma$, the PDE problem admits a unique solution u in a Hilbert space V. Moreover

$$\|u(\mathbf{y})\|_V \leq C(\mathbf{y}) \|\mathcal{F}(\mathbf{y})\|_{V'}.$$

Then, the PDE induces a map $u = u(\mathbf{y}) : \Gamma \to V$. The uniqueness assumption is essential for the forward uncertainty propagation problem to be well posed.

8.2.1 Forward Uncertainty Quantification tasks

Forward uncertainty quantification (UQ) encompasses a wide class of problems related with the evaluation of the impact of uncertainty (or variability) in the model inputs on the model outputs. In the context of mathematical models (and, more specifically, PDE models), forward UQ usually implies the evaluation of some *statistics of the solution*, such as:

- pointwise expected value: $\bar{u}(x) = \mathbb{E}[u(x,\cdot)]$, $x \in D$
- pointwise variance: $\mathrm{Var}[u](x) = \mathbb{E}\left[(u(x,\cdot) - \bar{u}(x))^2\right](x)$
- two points correlation: $Cov_u(x_1, x_2) = \mathbb{E}[(u(x_1,\cdot) - \bar{u}(x_1))(u(x_2,\cdot) - \bar{u}(x_2))]$

Another common task is to compute statistics of *Quantities of Interest* (expressed as functionals of the solution)

$$\Psi(\mathbf{y}) = Q(u(\mathbf{y})), \qquad \text{where } Q : V \to \mathbb{R}.$$

Usually, Ψ is a real-valued function of the random vector $\mathbf{y}$ and we would like to approximate its law. Typical examples of quantities of interest are:

$$Q(u) = u(\bar{x}) \qquad \text{for a given } \bar{x} \in D,$$

$$Q(u) = \sup_{x \in D} u(x),$$

$$Q(u) = \int_{\Sigma \subset D} f(u(x), \nabla u(x)) dx.$$

In particular, we may be interested in some failure probability event entailing the approximation of the probability

$$P(\Psi(\mathbf{y}) \geq \psi_{cr}) \quad \text{for some given critical value } \psi_{cr}.$$

Recall: in Section 3, we saw that either

- in case of a set of P uncertain, mutually independent parameters $\mathbf{y} = (y_1, \ldots, y_P) \in \mathbb{R}^P$ that may be represented as a random vector $\mathbf{y}(\omega) = (y_1(\omega), \ldots, y_P(\omega)) \in \mathbb{R}^P$, or
- in the case of random fields, considering the P-term truncation of the Karhunen-Loeve expansion of the field

we can obtain a finite-dimensional representation of random inputs in terms of P (possibly uncorrelated) random variables that indeed play the role of parameters.

The computational costs for UQ in the PDE setting increases with the number P of parameters used to model the uncertainty. That's clear. What we are not used to is that the number of random parameters P can be much larger than 2 or 3 (as it happens for spatial dimensions) and that the costs grow very, very quickly, even explosively, as P increases. As a result, many familiar numerical methods used for spatial discretization, e.g., for quadrature, become, in the UQ setting, too costly to use for approximation with respect to the random parameters. Here is the reason why, to face the so-called *curse of dimensionality*, suitable surrogate models or meta-models must be introduced to replace the expensive evaluation of the PDE (and related quantities of interest), or suitable multi-level (or multi-fidelity) techniques must be called into play.

8.3 Monte Carlo methods revisited

We start by revisiting the Monte Carlo method, showing how to estimate both the discretization and the statistical errors. Let $\mathbf{y} = (y_1, \dots, y_P)$ a random vector with density $f_y : \Gamma \to \mathbb{R}_+$, $u(\mathbf{y}) : \Gamma \to V$ a Hilbert-space valued function, $u \in L^2_\rho(\Gamma; V)$, and

$$Q : V \to \mathbb{R}$$

a continuous functional on V (possibly nonlinear), s.t. $\mathrm{E}[|Q(u(y))|^p] < \infty$ for p sufficiently large. Let us consider the problem of approximating

$$\mathbb{E}\left[Q(u(\mathbf{y}))\right]$$

and suppose that u is the solution of a PDE problem, depending on a set of random inputs; the formulation and the analysis will be sufficiently general – the only thing to keep in mind is that evaluate a sample $u(\mathbf{y}_m)$ might be an expensive task, from a numerical standpoint.

In the crude *Monte Carlo* method, we approximate expectations by sample averages: let $\{\mathbf{y}^{(n)}\}_{n=1}^N$ a sample of M iid replica of $\mathbf{y}$, and then approximate

$$\mathbb{E}\left[Q(u(\mathbf{y}))\right] \approx \frac{1}{N}\sum_{n=1}^{N} Q(u(\mathbf{y}^{(n)})).$$

Here for each $\mathbf{y}^{(n)}$ we have to solve the PDE to obtain $u(\mathbf{y}^{(n)})$, and then to evaluate the output QoI $Q(u(\mathbf{y}^{(n)}))$.

Remark 8.3.1. *On the one hand, Monte Carlo techniques allow ones to reuse deterministic legacy codes, and offer a convergence rate $\sqrt{N}$ resilient w.r.t. the dimension P of $\mathbf{y}$ (i.e., the parameter dimension) and the regularity of $u(\cdot)$; on the other hand, this convergence rate might be too low, and it is not possible to exploit regularity (when available).*

Assume now that the PDE has been discretized by some mean (finite elements, finite volumes, spectral methods, ...), so that in practice we compute discrete solutions $u_h(\mathbf{y}^{(n)})$, $n = 1, \dots, N$. Then our estimator is

$$\mathbb{E}\left[Q(u(\mathbf{y}))\right] \approx \hat{\mu}_h^{MC} := \frac{1}{N}\sum_{n=1}^{N} Q(u_h(\mathbf{y}^{(n)})).$$

Note that we can split the error as follows:

$$\mathbb{E}\left[Q(u(\mathbf{y}))\right] - \hat{\mu}_h^{MC} = \underbrace{\mathbb{E}\left[Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))\right]}_{\text{discretization error } \mathcal{E}^Q(h) \text{ (bias)}} + \underbrace{\mathbb{E}\left[Q(u_h(\mathbf{y}))\right] - \frac{1}{N}\sum_{n=1}^{N} Q(u_h(\mathbf{y}^{(n)}))}_{\text{statistical error } \mathcal{E}_h^Q(N)}$$

Let us now estimate the two error contributions; we will then further explore the. case of an elliptic PDE with random coefficients. Indeed, each PDE problem would require a different treatment, however the key points are to obtain a *stability estimate* on the PDE solution u (and its finite-dimensional approximation u_h), require a mild regularity assumption on the output QoI, and exploit general results from the Galerkin finite element method.

Remark 8.3.2. *Note that $\hat{\mu}_h^{MC}$ is a* biased *estimator, since $\mathbb{E}\left[\hat{\mu}_h^{MC}\right] = \mathbb{E}\left[Q(u_h(\mathbf{y}))\right] \neq \mathbb{E}\left[Q(u(\mathbf{y}))\right]$.*

8.3.1 Discretization error

Assume that the functional $Q : V \mapsto \mathbb{R}$, with $Q(0) = 0$, is globally Lipschitz, i.e. there exists a constant $C_Q > 0$ such that

$$|Q(u) - Q(v)| \leq C_Q \|u - v\|_V \qquad \forall u, v \in V.$$

Moreover, we assume that there exists $\alpha > 0$ and $C_u(\mathbf{y}) > 0$, with $\int_\Gamma C_u(\mathbf{y})^p \rho(\mathbf{y}) dy < \infty$ for some $p > 1$, such that

$$\|u(\mathbf{y}) - u_h(\mathbf{y})\|_V \leq C_u(\mathbf{y}) h^\alpha \qquad \forall \mathbf{y} \in \Gamma \text{ and } 0 < h < h_0. \tag{8.3}$$

Then, we can estimate the *discretization error* as

$$|\mathcal{E}^Q(h)| = \quad |\mathbb{E}\left[Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))\right]| = \left| \int_\Gamma (Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))) f_y(\mathbf{y}) d\mathbf{y} \right|$$

$$\leq \quad C_Q \int_\Gamma \|u(\mathbf{y}) - u_h(\mathbf{y})\|_V f_y(\mathbf{y}) d\mathbf{y} \leq C_Q h^\alpha \int_\Gamma C_u(\mathbf{y}) f_y(\mathbf{y}) d\mathbf{y} \leq C_Q \|C_u\|_{L^p_{f_y}} h^\alpha.$$

Usually $C_u(\mathbf{y})$ depends on some stronger norm $\|u(\mathbf{y})\|_W$. For instance, for $\mathbb{P}_1$ finite elements and $V = H^1(D)$, then one has $\alpha = \min\{s - 1, 1\}$ and $W = H^s(D)$, $s > 1$.

$$|\mathcal{E}^Q(h)| = |\mathbb{E}\left[Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))\right]| \leq C_Q \|C_u\|_{L^p_{f_y}} h^\alpha.$$

The case of an elliptic PDE with random coefficient (advanced level)

Let us now particularize the last error bound above in the case of the model problem of Section 8.2, dealing with an elliptic problem with uniformly bounded random coefficients,

$$\begin{cases} -\text{div}(a(x, \mathbf{y}) \nabla u(x, \mathbf{y})) = f(x) & x \in D, \\ u(x, \mathbf{y}) = 0 & x \in \partial D \end{cases}$$

with $\mathbf{y} \in \Gamma \subset \mathbb{R}^P$, where $D \subset \mathbb{R}^d$ is a convex, Lipschitz domain and $f \in L^2(D)$. As model for the random coefficient, we consider (with $P \leq \infty$)

$$a(x, \mathbf{y}) = \bar{a} + \sum_{i=1}^{P} \sqrt{\lambda_i} y_i b_i(x)$$

where $y_i \sim \mathcal{U}(-\sqrt{3}, \sqrt{3})$ i.i.d (so. that $\Gamma = [-\sqrt{3}, \sqrt{3}]^P$), $b_i \in C^\infty(D)$ and

$$\sum_{i=1}^{P} \sqrt{3\lambda_i} \|b_i\|_\infty \leq \delta \bar{a}, \qquad \delta \in (0, 1)$$

so that

$$(1 - \delta)\bar{a} \leq a(x, \mathbf{y}) \leq (1 + \delta)\bar{a} \qquad \forall \mathbf{y} \in \Gamma.$$

Denote $H^1_0(D) = \{v \in H^1(D),\ v = 0 \text{ on } \partial D\}$ endowed with the norm $\|v\|_{H^1_0(D)} = \|\nabla v\|_{L^2(D)}$.

Under the previous assumptions, thanks to the Lax-Milgram theorem, there exists a unique solution $u(\mathbf{y}) \in H^1_0(D)$, such that the following stability estimate holds:

$$\|u(\mathbf{y})\|_{H^1_0(D)} \leq \frac{C_{Poi}}{(1 - \delta)\bar{a}} \|f\|_{L^2(D)} \qquad \forall \mathbf{y} \in \Gamma$$

where C_{Poi} is the Poincaré constant, i.e.

$$\|v\|_{L^2(D)} \leq C_{Poi} \|v\|_{H^1_0(D)} \qquad \forall v \in H^1_0(D).$$

Remark 8.3.3. *It is possible to prove that if $u \in H_0^1(D)$ is a weak solution of an elliptic PDE in D, then $u \in H^2_{loc}(D)$ under very mild conditions on the coefficients. A weak solution $u \in H_0^1(D)$ remains smooth up to the boundary provided that the boundary of D is smooth and $a(\cdot, \mathbf{y}) \in C^1(\bar{D})$ for every $\mathbf{y} \in \Gamma$. Then $u \in H^2(D)$ and the following estimate holds:*

$$\|u\|_{H^2(D)} \leq \tilde{C}_2 \left(\|f\|_{L^2(D)} + \|u\|_{L^2(D)} \right).$$

Hence, provided that D is a regular domain, if $\|\nabla a(\cdot, \mathbf{y})\|_{L^\infty(D)} \leq C_a$, $\forall \mathbf{y} \in \Gamma$, we can conclude that

$$\exists\, C_2 > 0 \text{ s.t. } \|u(\mathbf{y})\|_{H^2(D)} \leq C_2 \qquad \forall \mathbf{y} \in \Gamma,$$

that is, we obtain a uniform bound in $\mathbf{y}$ on the H^2-norm of the solution.

We consider now a piecewise *linear* finite element approximation: for any $\mathbf{y} \in \Gamma$, find $u_h(\mathbf{y}) \in V_h$ s.t.

$$\int_D a(\cdot, \mathbf{y}) \nabla u_h(\mathbf{y}) \cdot \nabla v_h(\mathbf{y}) = \int_D f v_h \qquad \forall v_h \in V_h$$

where $V_h \subset H_0^1(D)$ is the space of continuous piecewise linear functions on a triangulation T_h, vanishing on ∂D. The discrete solution $u_h(\mathbf{y})$ satisfies the same stability estimate as the continuous one, that is,

$$\|u_h(\mathbf{y})\|_{H_0^1(D)} \leq \frac{C_{Poi}}{(1-\delta)\bar{a}} \|f\|_{L^2(D)} \qquad \forall \mathbf{y} \in \Gamma$$

and, since $Q(\cdot)$ is Lipschitz,

$$|Q(u_h(\mathbf{y}))| \leq C_Q \|u_h(\mathbf{y})\|_{H_0^1(D)}$$

so that

$$\mathbb{E}\left[|Q(u_h(\mathbf{y}))|^p\right]^{1/p} \leq \mathbb{E}\left[\left(C_Q \|u_h(\mathbf{y})\|_{H_0^1(D)}\right)^p\right]^{1/p} \leq \frac{C_Q C_P}{(1-\delta)\bar{a}} \|f\|_{L^2(D)}$$

for each $p \geq 1$, thanks to the stability estimate fulfilled by $u_h(\mathbf{y}$. Then, from the standard theory of the Galerkin finite element method, we have that

$$\|u(\mathbf{y}) - u_h(\mathbf{y})\|_{L^2(D)} + h \|\nabla u(\mathbf{y}) - \nabla u_h(\mathbf{y})\|_{L^2(D)} \leq C |u(\mathbf{y})|_{H^2(D)} h^2.$$

Since the H^2 bound is uniform in $\mathbf{y}$, all moments are bounded and (8.3) holds:

$$\mathbb{E}\left[\|u(\mathbf{y}) - u_h(\mathbf{y})\|^p_{L^2(D)}\right]^{1/p} + h\mathbb{E}\left[\|\nabla u(\mathbf{y}) - \nabla u_h(\mathbf{y})\|^p_{L^2(D)}\right]^{1/p} \leq C h^2$$

for each $p \geq 1$. Hence, for a Lipschitz functional $Q : H_0^1(D) \to \mathbb{R}$, depending on the solution of an elliptic problem with uniformly bounded random coefficients, we have shown that

$$\mathbb{E}\left[Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))\right] \leq C_Q \mathbb{E}\left[\|u(\mathbf{y}) - u_h(\mathbf{y})\|^p_{H_0^1(D)}\right] \leq C_Q C_u h.$$

8.3.2 Statistical error

Let us analyze the error assuming that Q is a globally Lipschitz functional. Observe now that we have

$$\mathcal{E}_h^Q(N) = \mathbb{E}\left[Q(u_h(\mathbf{y}))\right] - \frac{1}{N}\sum_{n=1}^N Q(u_h(\mathbf{y}^{(n)})) = \frac{1}{N}\sum_{n=1}^N \left(\mathbb{E}\left[Q(u_h(\mathbf{y}))\right] - Q(u_h(\mathbf{y}^{(n)}))\right)$$

and $\mathbb{E}\left[\mathcal{E}_h^Q(N)\right] = 0$ (here expectation is with respect to the random sample). Then

$$\mathrm{Var}\left[\mathcal{E}_h^Q(N)\right] = \frac{1}{N}\mathrm{Var}\left[Q(u_h(\mathbf{y}))\right].$$

Recall now that (8.3), that is,

$$\|u(\mathbf{y}) - u_h(\mathbf{y})\|_V \leq C_u(\mathbf{y})h^\alpha \qquad \forall \mathbf{y} \in \Gamma \text{ and } 0 < h < h_0.$$

Then, we have

$$\|u(\mathbf{y}) - u_h(\mathbf{y})\|_V^2 \leq C_u^2(\mathbf{y})h^{2\alpha}$$

and taking expectations,

$$\mathbb{E}\left[\|u(\mathbf{y}) - u_h(\mathbf{y})\|_V^2\right] \leq h^{2\alpha}\mathbb{E}\left[C_u^2(\mathbf{y})\right]$$

that is,

$$\|u - u_h\|^2_{L^2_{f_y}(\Gamma;V)} \leq h^{2\alpha}\|C_u\|_{L^2_{f_y}(\Gamma)}.$$

Hence, by triangular inequality, we can estimate

$$\begin{aligned}\mathrm{Var}\left[Q(u_h(\mathbf{y}))\right] \leq\ & \mathrm{E}[Q(u_h(\mathbf{y}))^2] \leq C_Q^2\|u_h\|^2_{L^2_{f_y}(\Gamma;V)} \\ \leq\ & 2C_Q^2\left(\|u\|^2_{L^2_{f_y}(\Gamma;V)} + \|u - u_h\|^2_{L^2_{f_y}(\Gamma;V)}\right) \leq 2C_Q^2\left(\|u\|^2_{L^2_{f_y}(\Gamma;V)} + h^{2\alpha}\|C_u\|_{L^2_{f_y}(\Gamma)}\right) \\ <\ & +\infty\end{aligned}$$

Hence, we have proved that, for Lipschitz functionals Q, we can bound the statistical error as we did in Chapter 4, so that, asymtptotically ($N \to \infty$) with probability $1-\alpha$,

$$|\mathcal{E}_h^Q(N)| = \left|\mathbb{E}\left[Q(u_h(\mathbf{y}))\right] - \frac{1}{N}\sum_{n=1}^{N} Q(u_h(\mathbf{y}^{(n)}))\right| \leq z_{1-\alpha/2}\frac{\sqrt{\mathrm{Var}\left[Q(u_h)\right]}}{\sqrt{N}}$$

Moreover, we have that the mean square error[2] of the MC estimator (which coincides with its variance only if the estimator is unbiased, which is not the case at hand) is

$$\begin{aligned}\mathrm{MSE}(\hat{\mu}_h^{MC}) :=\ & \mathbb{E}\left[(\hat{\mu}_h^{MC} - \mathbb{E}\left[Q(u(\mathbf{y}))\right])^2\right] = (\mathbb{E}\left[\hat{\mu}_h^{MC}\right] - \mathbb{E}\left[Q(u(\mathbf{y}))\right])^2 + \mathrm{Var}\left[\hat{\mu}_h^{MC}\right] \\ =\ & (\mathbb{E}\left[Q(u(\mathbf{y})) - Q(u_h(\mathbf{y}))\right])^2 + \frac{\mathrm{Var}\left[Q(u_h(\mathbf{y}))\right]}{N}\end{aligned}$$

that is, can be split as the sum of the discretization error and of the MC error.

8.3.3 Monte Carlo complexity analysis

In the previous section we have seen that

$$\mathbb{E}\left[Q(u(\mathbf{y}))\right] - \hat{\mu}_h^{MC} = \mathcal{E}^Q(h) + \mathcal{E}_h^Q(N) \leq C_{discr}h^\alpha + \frac{C_{stat}}{\sqrt{N}}.$$

we highlight h and N because they are the two "ingredients" that we need if we want to quantify how much expensive the MC estimation is.

[2] Recall that the mean squared error (MSE) of an estimator measures the average of the squares of the errors, that is, it is the second moment of the error. For an unbiased estimator, the MSE is the variance of the estimator. If $\hat{\theta}$ is the estimator of θ, then

$$MSE(\hat{\theta}) := \mathbb{E}\left[(\theta - \hat{\theta})^2\right].$$

Alternatively,

$$\mathbb{E}\left[(\theta - \hat{\theta})^2\right] = \mathbb{E}\left[\hat{\theta}^2\right] + \mathbb{E}\left[\theta^2\right] - 2\theta\mathbb{E}\left[\hat{\theta}\right] = \mathrm{Var}\left[\hat{\theta}\right] + (\mathbb{E}\left[\hat{\theta}\right])^2 + \theta^2 - 2\theta\mathbb{E}\left[\hat{\theta}\right] = \mathrm{Var}\left[\hat{\theta}\right] + (\mathbb{E}\left[\hat{\theta}\right] - \theta)^2 = \mathrm{Var}\left[\hat{\theta}\right] + \mathrm{Bias}^2(\hat{\theta}).$$

ABSTRACT COMPLEXITY THEOREM for standard Monte Carlo.

Assume that $\exists\, \alpha, \gamma > 0$ such that:

1. $|\mathbb{E}[Q_h - Q]| = O(h^\alpha)$ as $h \to 0$ (h = mesh size) ($Q_h = Q(u_h(\vec{y}^{(i)}))$, $Q = Q(u(\vec{y}^{(i)}))$)
2. $\text{cost}(Q_h^{(i)}) = O(h^{-d\gamma})$, where γ depends on the numerical solver is the cost to compute a sample from the approximation Q_h and d is the spatial dimension (in case of PDE problem, $d=1$ for ODE)

Then, $\forall \varepsilon > 0$ the total cost to compute a standard MC estimator for $\mathbb{E}[Q]$ such that MSE $< \varepsilon^2$ satisfies:

$$\text{cost}(\hat{Q}_h) = O\left(\varepsilon^{-\left(2+\frac{d\gamma}{\alpha}\right)}\right)$$

↳ the cost of the estimator scales as a given power of ε

proof.

Let us assume now that the computational work to solve for each $u_h(\mathbf{y}^{(n)})$, a d-dimensional problem in $\mathbb{R}^d$, is

$$C_h = O(h^{-d\gamma}), \quad = \text{cost of a single sample } Q_h^{(i)}$$

where usually $\gamma \in [1, 3]$ depending on the numerical solver. For instance, $\gamma = 3$ for a direct solver and a full matrix, while $\gamma \approx 1$ for an optimal solver with linear complexity (or a direct solver with a sparse matrix[3]). We have therefore the following estimates:

$$\text{Total work}: W \propto N h^{-d\gamma}$$

← In MC we have to compute N times $Q_h^{(i)}$ and the cost of one is $h^{-d\gamma}$

$$\text{Total error}: |\mathcal{E}^Q(h)| + |\mathcal{E}_h^Q(N)| \le C_{discr} h^\alpha + \frac{C_{stat}}{\sqrt{N}}.$$

To choose optimally h and N, we can minimize the work subject to an accuracy constraint:

$$\begin{cases} N h^{-d\gamma} \to \min \\ \text{s.t.} \\ C_{discr} h^\alpha + \dfrac{C_{stat}}{\sqrt{N}} \le \text{tol}. \end{cases} \tag{8.4}$$

we minimize the total cost under a constraint (error $< \varepsilon$); but we already estimated the error

Equivalently, we might require that $\text{MSE}(\hat{\mu}_h^{MC}) \le \text{tol}^2$. ⟺ $\mathbb{E}[Q(u(\vec{y}))] - \hat{Q}_h < \varepsilon$

The Lagrangian of problem (8.4) is

$$\mathcal{L}(N, h, \lambda) = N h^{-d\gamma} + \lambda \underbrace{\left(C_{discr} h^\alpha + \frac{C_{stat}}{\sqrt{N}} - \text{tol}\right)}_{\text{constraint}}$$

Lagrangian = cost + λ constraint

Enforcing

$$\partial_N \mathcal{L}(N, h, \lambda) = \partial_h \mathcal{L}(N, h, \lambda) = 0$$

yields

$$\frac{1}{\sqrt{N}} = \frac{2\alpha C_{discr}}{d\gamma C_{stat}} h^\alpha \quad \Rightarrow \quad N = O(h^{-2\alpha}). \tag{8.5}$$

Indeed,

- $\partial_N \mathcal{L}(N, h, \lambda) = h^{-d\gamma} - \frac{1}{2}\lambda C_{stat} N^{-3/2} = 0 \quad \Rightarrow \quad h^{-d\gamma} = \frac{1}{2}\lambda C_{stat} N^{-3/2}$

and

- $\partial_h \mathcal{L}(N, h, \lambda) = -d\gamma N h^{-d\gamma - 1} + \alpha\lambda h^{\alpha-1} C_{discr} = 0$

so that

$$0 = -d\gamma N h^{-d\gamma-1} + \alpha\lambda h^{\alpha-1} C_{discr} = -\frac{d\gamma N}{h}\left(\frac{1}{2}\lambda C_{stat} N^{-3/2}\right) + \alpha\lambda h^{\alpha-1} C_{discr} = 0$$

whence

$$\frac{1}{\sqrt{N}} \frac{d\gamma\lambda C_{stat}}{2h} = \alpha\lambda h^{\alpha-1} C_{discr} \quad \Rightarrow \quad \frac{C_{stat}}{\sqrt{N}} = \frac{2h}{d\gamma\lambda} \alpha\lambda h^{\alpha-1} C_{discr} = \frac{2\alpha C_{discr}}{d\gamma} h^\alpha.$$

Using the accuracy constraint we have

$$C_{discr} h^\alpha + \frac{C_{stat}}{\sqrt{N}} = C_{discr} h^\alpha + \frac{2\alpha C_{discr}}{d\gamma} h^\alpha = \text{tol}$$

whence

$$C_{discr}\left(1 + \frac{2\alpha}{d\gamma}\right) h^\alpha = \text{tol} \quad \Rightarrow \quad h^\alpha = \text{tol} \frac{1}{C_{discr}(1 + 2\alpha/(d\gamma))} \quad \Rightarrow \quad h = O(\text{tol}^{1/\alpha}). \tag{8.6}$$

[3] Remember: for a matrix $A \in \mathbb{R}^{M \text{ times } M}$, a direct solver relying on either the LU or the Cholesky factorization requires $O(M^3)$ operations, the Thomas solver for a tridiagonal (hence, sparse) matrix requires $O(M)$ operations, while iterative solvers can require $O(M^2)$ operations. A sparse direct solver on the other hand, such as the one provided by Matlab through the backslash operation, will have at worst $\gamma = 1.5$.

Finally, using (8.5)-(8.6), we obtain that

$$W \propto Nh^{-d\gamma} \propto h^{-2\alpha}h^{-d\gamma} = h^{-2\alpha-d\gamma} \propto (\text{tol}^{1/\alpha})^{-2\alpha-d\gamma} = \text{tol}^{-(2+d\gamma/\alpha)},$$

that is,

The resulting complexity (error versus computational work) of the Monte Carlo method is then

$$W_{MC} \propto \text{tol}^{-(2+d\gamma/\alpha)}.$$

d, γ, α depend on the problem

8.3.4 Control Variates revisited

We have seen in Chapter 5 how we can improve on Monte Carlo estimation. In particular, one of the classic approaches to Monte Carlo variance reduction is through the use of a control variate. In the following sections, we will also see how a suitable use of control variates, further generalized, allows us to enhance computational efficiency. Before to do that, let us revisit the control variate method.

Notice that we already did some analysis like this in the Lodka-Volterra (predator-prey) example : we had that $W_{MC} \propto \varepsilon^{-3}$ and this is because in those settings : $d\gamma/\alpha = 1$ (d = dimension = 1, α = order of the explicit Euler = 1, γ = 1)

Suppose we wish to estimate $\mu = \mathbb{E}[Z]$, and there is a control variate Y which is well correlated to Z and has a known expectation $\mathbb{E}[Y]$. In that case, we have seen that we can use the following unbiased estimator for $\mathbb{E}[Z]$ based on N independent samples:

$$\hat{\mu}_{CV} = \frac{1}{N}\sum_{n=1}^{N}\left(Z^{(n)} - \beta Y^{(n)}\right) + \beta\mathbb{E}[Y]$$

where the optimal value is $\beta = \beta_{opt}$ given by

$$\beta_{opt} = -\frac{Cov(Z,Y)}{\text{Var}[Y]};$$

in this case, the variance of the control variate estimator is reduced by a factor $(1 - Corr(Z,Y)^2)$ compared to the standard estimator. We remark that:

- we always obtain a variance reduction as long as Y is correlated with Z;
- in some cases (see e.g. Multi Level Monte Carlo), if one knows a priori that Y is strongly positively (resp. negatively) correlated with Z , the choice $\beta = 1$ (resp. $\beta = -1$) works well.

Does control variates pay in terms of computational work?

Assume that the work to generate the pair $(Z^{(j)}, Y^{(j)})$ is a factor $(1+\theta)$ times the work to generate $Z^{(j)}$. Hence we have that, to ensure an error between $\mathbb{E}[Z]$ and its MC estimate below tol, we need to choose

$$N \geq \left(\frac{C}{\text{tol}}\right)^2 \text{Var}[Z]$$

where $C = 4z^2_{1-\alpha/2} > 0$. This fact can be obtained imposing that the width of the $(1-\alpha)$ asymptotic CI is less than tol.

Hence, denoting by $\tilde{Z}_{\beta_{opt}} = Z - \hat{\beta}_{opt}(Y - \mathbb{E}[Y])$ we have that

method	# samples	computational work
MC	$\left(\frac{C}{\text{tol}}\right)^2 \text{Var}[Z]$	$\left(\frac{C}{\text{tol}}\right)^2 \text{Var}[Z]$
MC + Control Variate	$\left(\frac{C}{\text{tol}}\right)^2 \text{Var}\left[\tilde{Z}_{\beta_{opt}}\right]$	$(1+\theta)\left(\frac{C}{\text{tol}}\right)^2 \text{Var}\left[\tilde{Z}_{\beta_{opt}}\right]$

and the control variate strategy only pays if

$$\mathrm{Var}\,[Z] > (1+\theta)\,\mathrm{Var}\left[\tilde{Z}_{\beta_{opt}}\right]$$

Recalling that $\mathrm{Var}\left[\tilde{Z}_{\beta_{opt}}\right] = \mathrm{Var}\,[Z]\,(1-Corr(Z,Y)^2)$ (see Section 5.3) we have that the condition above is satisfied if and only if

$$1 > (1+\theta)(1-Corr(Z,Y)^2) \quad \text{that is,} \quad Corr(Z,Y)^2 > \frac{\theta}{1+\theta}.$$

Hence, when applying the control variate method to estimate output quantities of interest related with mathematical models whose numerical approximation might be expensive, the increase of cost entailed by the additional evaluation of model Y must fulfill the relation above to make the method computationally feasible.

8.4 Multi level Monte Carlo

Let us now apply the idea of the control variate (CV) method to the case of the estimation of $\mathbb{E}\,[Q(u_h(\mathbf{y}))]$, where $u_h(\mathbf{y})$ is the numerical approximation of the solution to a given mathematical model, depending on random inputs $\mathbf{y}$. For the sake of notation, let us denote by

$$P = Q(u(\mathbf{y})), \qquad P_h = Q(u_h(\mathbf{y}))$$

To apply the control variate method, let us consider a random variable Y, correlated with P_h, and with *known mean*. Then, we consider the MC estimator on

$$P_{h,Y} = P_h - \beta(Y - \mathbb{E}\,[Y])$$

(notice that $\mathbb{E}\,[P_{h,Y}] = \mathbb{E}\,[P_h]$) so that the CV estimator reads as

$$\hat{\mu}_h^{CV} = \frac{1}{N}\sum_{n=1}^{N}(P_h^{(n)} - \beta Y^{(n)}) + \beta\mathbb{E}\,[Y]\,.$$

Two possible ideas to choose Y can be:

- use a surrogate model $Y = P^{surr}$ with numerically optimized β; this is the behind a recent generalization of the MC method, which goes under the name of *multi-fidelity Monte Carlo*;
- use a coarser discretization, by taking, e.g., $Y = P_{2h}$, usually with $\beta = 1$. This is the idea of the *two level Monte Carlo*, that we explore and generalize below.

In both these cases, $\mathbb{E}\,[Y]$ is not known, in general, so that it must be estimated with an independent MC using a larger sample size (since Y is in general a cheaper problem to solve). This is the key point connecting the control variate method and the two-level Monte Carlo method.

For the sake of notation, let us denote the problem P_h by P_1 and the control variate Y by P_0. Here P_1 and P_0 represent two models arising from the discretization of the problem, obtained for two different *levels* $h_1 < h_0$ - here h represents a discretization parameter, such as time or space discretization. Hence, we apply MC on

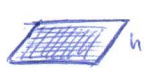
$$P_{1,0} := P_1 - (P_0 - \mathbb{E}\,[P_0])$$

where we have taken $\beta = 1$ in the general definition of the control variate method. Notice that

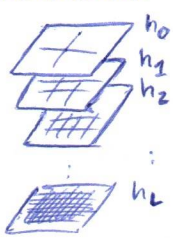
$$\mathbb{E}\,[P_{1,0}] = \mathbb{E}\,[P_0] + \mathbb{E}\,[P_1 - P_0] = \mathbb{E}\,[P_1]\,.$$

based on averaging the knowledge of different models, namely: the idea is to estimate the expectation of a quantity of interest $\mathbb{E}[Q_{h_L}(u(\vec{y}))]$ (since $\mathbb{E}[Q(u(\vec{y}))]$ is unaccessible) by splitting this evaluation on a sequence of levels (= sequence of models that are characterized by different values of discretization parameters $(h_0, h_1, \dots, h_L)$, where h_L is the finest and most expensive level and h_0 is the most coarse one)).

To "see" what we are doing: suppose we want:



with crude MC we would just discretize like this and we would do all the samples of $Q_{h_L}^{(i)}$ on this. However this is too expensive. Instead we divide in levels:



and we evaluate some samples on each level. In this way we can get the same result (in terms of variance, e.g.) for a smaller cost (since evaluating $Q_h^{(i)}$ on h_j with $j \neq L$ is cheaper since the model is more coarse)

Thus, we obtain the following CV estimator:

$$\hat{\mu}_h^{CV} = \frac{1}{N_1}\sum_{n=1}^{N_1}(P_1^{(n)} - P_0^{(n)}) + \mathbb{E}[P_0]$$

This latter quantity can then be estimated by the following two-level estimator

two level and control variates are somehow equivalent

$$\hat{\mu}_1^{MLMC} := \frac{1}{N_1}\sum_{n=1}^{N_1}(P_1^{(n)} - P_0^{(n)}) + \frac{1}{N_0}\sum_{n=1}^{N_0} P_0^{(n)}, \qquad N_0 > N_1.$$

In other words, P_0 plays the role of *control variate* for P_1. The two key differences from the control variate approach are that the value of $\mathbb{E}[P_0]$ is not known, so has to be estimated, and we use $\beta = 1$. Here $P_1^{(n)} - P_0^{(n)}$ represents the difference between P_1 and P_0 for the same stochastic sample, so that $P_1^{(n)} - P_0^{(n)}$ is small and has small variance; the precise construction depends on the application and some examples will be shown later.

we expect this to be small since it's the same quantity of interest evaluated on two solutions that differ only for the mesh grid size (we have different discretizations for P_1 and P_0, that's the only difference)

this is because the models are highly correlated

If we define C_0 and C_1 to be the cost of computing a single sample of P_0 and $P_1 - P_0$, respectively, then the total cost is $N_0C_0 + N_1C_1$, and if V_0 and V_1 are the variances of P_0 and $P_1 - P_0$, then the overall variance is $N_0^{-1}V_0 + N_1^{-1}V_1$, assuming that

$C_0 = \text{cost}(P_0^{(n)})$
$C_1 = \text{cost}(P_1^{(n)} - P_0^{(n)})$

$$\frac{1}{N_0}\sum_{n=1}^{N_0} P_0^{(n)} \quad \text{and} \quad \frac{1}{N_1}\sum_{n=1}^{N_1}(P_1^{(n)} - P_0^{(n)})$$

use independent samples. Hence, treating the integers N_0, N_1 as real variables and performing a constrained minimization using a Lagrange multiplier, the variance is minimized for a fixed cost by choosing

$$\frac{N_1}{N_0} = \frac{\sqrt{V_1}/C_1}{\sqrt{V_0/C_0}}.$$

> The use of control variates can be seen as a two-level Monte Carlo method, in which two different *models* are used to improve the error in the Monte Carlo estimate. The generalization of this idea is the so-called Multi level Monte Carlo (MLMC) method, and has been used successfully with PDEs, in which the different models are generated by considering further levels of discretization, and a coarse numerical approximation acts as a control variate of a fine one.

the idea is to use Q_h on different levels: $h_0 > h_1 > \dots > h_L$ of the discretization parameter ($P_L = Q_{h_L}$) *

The multi level generalization is quite natural: take a sequence $P_0, \dots, P_L$ of models where $P_0, \dots, P_{L-1}$ approximate P_L with increasing accuracy, but also increasing *cost*. These models might correspond to a sequence of finer and finer discretizations $h_0 > h_1 > \dots h_L$, and we assume that mesh size h_L achieves the desired accuracy. Here we denote by

$$P_\ell \equiv P_{h_\ell} = Q(u_{h_\ell}(\mathbf{y})), \qquad \ell = 0, \dots, L.$$

quantity of interest evaluated on the solution obtained numerically at the level ℓ (h_ℓ)

the desired model. Here the goal is then to approximate $\mathbb{E}[P]$ by $\mathbb{E}[P_L]$, and this latter quantity by combining the knowledge of the models $P_0, \dots, P_{L-1}$. Then, considering a sequence of sample sizes $M_0 > M_1 > \dots > M_L$, we obtain the following unbiased estimator[4] for $\mathbb{E}[P_L]$:

[4] Somehow, we are expressing

$$\mathbb{E}[P_L] = \mathbb{E}[P_0] + \sum_{\ell=1}^{L}\mathbb{E}[P_\ell - P_{\ell-1}]$$

and estimating the right-hand side by sampling independently each term.

notice that for simplicity it usually is that: $h_{\ell-1} = m \cdot h_\ell$ (namely there is a uniform grid refined)

→ sometimes this is called **geometric MLMC**

* $P_L = Q_{h_L}$ is an approximation of $Q(u(y^*))$ build on a mesh of level h_L.

Q_{h_L}

The Multi Level Monte Carlo estimator for $\mathbb{E}[P_L]$ is then given by

$$\hat{\mu}_L^{MLMC} = \frac{1}{N_0}\sum_{n=1}^{N_0} P_0^{(0,n)} + \sum_{\ell=1}^{L}\frac{1}{N_\ell}\sum_{n=1}^{N_\ell}\left(P_\ell^{(\ell,n)} - P_{\ell-1}^{(\ell,n)}\right)$$

3 properties

with the inclusion of the level ℓ in the superscript (ℓ, n) indicating that independent samples are used at each level of correction. In other words, we consider independent samples $\{\mathbf{y}_\ell^{(n)}, n = 1, \dots, N_\ell\}$ of i.i.d. replica on each level. Note that

1. $\mathrm{E}[\hat{\mu}_L^{MLMC}] = \mathrm{E}[P_L]$.

for every level we resample (otherwise it seems like we add and subtract the same thing)

the estimator is unbiased (w.r.t. the numerical quantity that we wanted to estimate) (namely: $\mathbb{E}[\hat{\mu}_L^{MLMC}] = \mathbb{E}[Q_{h_L}] \neq \mathbb{E}[Q]$ because there is a discretization error)

The variance of the MLMC estimator, thanks to the independent samples among levels, is

2. $$\mathrm{Var}\left[\hat{\mu}_L^{MLMC}\right] = \frac{\mathrm{Var}[P_0]}{N_0} + \sum_{\ell=1}^{L}\frac{\mathrm{Var}[P_\ell - P_{\ell-1}]}{N_\ell}$$

The key point is that since $\mathrm{Var}(g_\ell - g_{\ell-1})$ gets smaller and smaller for ℓ large, one can take N_ℓ smaller and smaller, so that only few replica on the fine grid (with mesh size h_L) are required.

Remark 8.4.1. *Moreover, we have that the mean squared error of* $\hat{\mu}_L^{MLMC}$ *is*

3. $$MSE(\hat{\mu}_L^{MLMC}) = \mathbb{E}\left[(\hat{\mu}_L^{MLMC} - \mathbb{E}[P])^2\right] = \left(\mathbb{E}\left[\hat{\mu}_h^{MLMC} - \mathbb{E}[P]\right]\right)^2 + \mathrm{Var}\left[\hat{\mu}_L^{MLMC}\right]$$

the expectation

$$= \boxed{\left(\mathbb{E}[P - P_L]\right)^2} + \boxed{\frac{\mathrm{Var}[P_0]}{N_0} + \sum_{\ell=1}^{L}\frac{\mathrm{Var}[P_\ell - P_{\ell-1}]}{N_\ell}}$$

(bias)^2 (note that this is the discretization of the finest level (h_L))

sample variance (variance of the MLMC estimator)

where $P - P_L = Q(u(\mathbf{y})) - Q(u_{h_L}(\mathbf{y}))$*; hence, the MSE is the sum of the variance of the MLMC estimator (statistical error), and of the discretization error at level* L.

8.4.1 Optimal choice of N_ℓ

How to select the number N_ℓ of replicas at each *level*? We can find the best possible strategy. If we denote by

- C_0 = cost of generating a realization of P_0;
- C_ℓ = cost of generating a realization of $P_\ell - P_{\ell-1}$, $\ell > 0$;
- $V_0 = \mathrm{Var}[P_0]$;
- $V_\ell = \mathrm{Var}[P_\ell - P_{\ell-1}]$, $\ell > 0$.

the overall cost (or work) and variance of the MLMC estimator are, respectively,

$$\text{Total work}: \; W_{MLMC} = \sum_{\ell=0}^{L} N_\ell C_\ell$$

at each level we need to evaluate N_ℓ times the model and C_ℓ is the cost of evaluating the model $P_\ell - P_{\ell-1}$

and

$$\text{Total variance}: \; \mathrm{Var}\left[\hat{\mu}_L^{MLMC}\right] = \sum_{\ell=0}^{L}\frac{V_\ell}{N_\ell}.$$

The problem is then to find optimal $\{M_\ell\}$ to minimize the cost at a fixed variance level[5] that is,

$$\sum_{\ell=0}^{L} N_\ell C_\ell \to \min_{\{N_\ell\}} \qquad \text{s.t.} \quad \sum_{\ell=0}^{L} N_\ell^{-1} V_\ell \leq \mathrm{tol}^2$$

[5] We could also minimize the variance at a given cost...

In which sense we can obtain **variance reduction**? (since control variates is a variance reduction technique). We want to select $h_0, \dots, h_L$ (in particular h_L) and the number of samples $N_0, \dots, N_L$ to balance the terms in MSE.

- As before, $h_L = O(\varepsilon^{1/\alpha})$
- Why do we get variance reduction? Or lower cost for the same variance?

As soon as we coarse the problem:

1. the cost per sample decrease rapidly provided that: $\mathrm{cost}(Q_h^{(n)}) = O(h^{-\delta})$ as $h \to 0$
2. since $Q_h \to Q$ as $h \to 0$ then: $\mathrm{Var}(Q_{h_\ell} - Q_{h_{\ell-1}}) \to 0$ at $\ell \to \infty$, allowing for smaller and smaller sample size N_ℓ on higher and higher levels.

(this gets smaller and smaller as we coarse the problem)

It remains: How to choose optimally N_ℓ at each level?

If we replace M_l by continuous variables, the optimal solution is

$$N_\ell = \text{tol}^{-2}\sqrt{\frac{V_\ell}{C_\ell}}\left(\sum_{j=0}^{L}\sqrt{V_j C_j}\right).$$

Indeed, we can write the Lagrangian function

$$\mathcal{L}(N_0,\dots,N_\ell,\lambda) = \sum_{\ell=0}^{L} N_\ell C_\ell - \lambda\left(\text{tol}^2 - \sum_{\ell=0}^{L}\frac{V_\ell}{N_\ell}\right)$$

Then,

$$\frac{\partial\mathcal{L}}{\partial N_\ell} = C_\ell - \lambda\frac{V_\ell}{N_\ell^2} = 0,$$

hence

$$N_\ell = \sqrt{\lambda\frac{V_\ell}{C_\ell}}.$$

Replacing in the constraint gives

$$\sum_{\ell=0}^{L}\frac{V_\ell}{N_\ell} = \text{tol}^2 \quad \Rightarrow \quad \sqrt{\lambda} = \text{tol}^{-2}\sum_{\ell=0}^{L}\sqrt{V_\ell C_\ell}.$$

In practice, one should take the ceiling of the real value N_ℓ (important if $N_\ell < 1$).

The optimal sample sizes of the samples at each level required by the evaiuation of the MLMC estimator $\hat{\mu}_L^{MLMC}$ are

$$N_\ell^* = \left\lceil \text{tol}^{-2}\sqrt{\frac{V_\ell}{C_\ell}}\left(\sum_{j=0}^{L}\sqrt{V_j C_j}\right)\right\rceil$$

while the total work required is

$$W_{MLMC} = \sum_{\ell=0}^{L} N_\ell^* C_\ell.$$

8.4.2 Geometric MLMC

The analysis can be further detailed in the case where mesh refinements are *geometric*, that is,

$$h_\ell = h_0\delta^\ell \qquad \text{for a given } \ 0 < \delta < 1.$$

Starting from the fact that

$$\text{MSE}(\hat{\mu}_L^{MLMC}) = \left(\mathbb{E}\left[\hat{\mu}_h^{MLMC} - \mathbb{E}\left[P\right]\right]\right)^2 + \text{Var}\left[\hat{\mu}_L^{MLMC}\right]$$

where

$$\hat{\mu}_L^{MLMC} = \sum_{\ell=0}^{L}\mathcal{A}_\ell, \qquad \mathcal{A}_\ell = \frac{1}{N_\ell}\sum_{n=1}^{N_\ell}\left(P_\ell^{(\ell,n)} - P_{\ell-1}^{(\ell,n)}\right)$$

with $P_{-1} = 0$, and

$$\mathbb{E}\left[\hat{\mu}_L^{MLMC}\right] = \mathbb{E}\left[P_L\right], \qquad \text{Var}\left[\hat{\mu}_L^{MLMC}\right] = \sum_{\ell=0}^{L} N_\ell^{-1}V_\ell, \qquad V_\ell = \text{Var}\left[P_\ell - P_{\ell-1}\right],$$

to ensure that MSE $\leq \text{tol}^2$, it is sufficient to ensure that $\text{Var}\left[\hat{\mu}_L^{MLMC}\right]$ and $\left(\mathbb{E}\left[\hat{\mu}_L^{MLMC}\right] - \mathbb{E}\left[P\right]\right)^2$ are both less than $\frac{1}{2}\text{tol}^2$.

Combining this idea with a geometric sequence of levels in which the cost increases exponentially with level, while both the weak error $|\mathbb{E}\left[P - P_\ell\right]$ and the multilevel correction variance V_ℓ decrease exponentially, leads to the following complexity analysis (error versus cost).

Assume for a problem in $\mathbb{R}^d$ (d-dimensional) that there exist independent estimators $\mathcal{A}_\ell$ based on N_ℓ Monte Carlo samples, each with expected cost C_ℓ and variance V_ℓ, such that

1. $h_\ell = h_0 \delta^\ell$ for a given $0 < \delta < 1$ (geometric meshes)
2. $|\mathbb{E}\left[P - P_\ell\right]| \leq C_w h_\ell^\alpha$ (weak rate of convergence)
3. $\mathbb{E}\left[(P - P_\ell)^2\right] \leq C_s h_\ell^\beta$ (strong rate of convergence)
4. $C_\ell = C_c h_\ell^{-d\gamma}$.

Notice that from the third point above, it follows that

$$V_\ell = \text{Var}(P_\ell - P_{\ell-1}) \leq \mathbb{E}\left[(P_\ell - P_{\ell-1})^2\right] \leq 2\mathbb{E}\left[(P - P_\ell)^2\right] + 2\mathbb{E}\left[(P - P_{\ell-1})^2\right] \leq 2C_s(\delta^\beta + 1)h_{\ell-1}^\beta$$

that is,

$$V_\ell \leq C_v h_{\ell-1}^\beta \qquad \text{with} \quad C_v = 2C_s(\delta^\beta + 1).$$

Moreover, one always has $\beta \leq 2\alpha$ (typically $\beta = 2\alpha$ for PDEs with random coefficients), since, by Cauchy-Schwarz inequality,

$$\mathbb{E}\left[P - P_\ell\right] \leq \mathbb{E}\left[(P - P_\ell)^2\right]^{1/2} \leq C_s^{1/2} h_\ell^{\beta/2}$$

hence, $\alpha \geq \beta/2$. The following, fundamental result holds (curious about the proof? Check **??**).

Theorem [Cliffe-Giles-Scheichl-Teckentrup 2011].
Under the assumptions 1-4 above, if $2\alpha \geq \min(\beta, d\gamma)$, the computational work required to approximate $\mathbb{E}\left[P\right]$ with MLMC with accuracy $0 < \text{tol} < 1/e$ in mean square sense, that is

$$\mathbb{E}\left[(\hat{\mu}_L^{MLMC} - \mathbb{E}\left[P\right])^2\right] \leq \text{tol}^2$$

is bounded as follows:

$$W_{MLMC} \approxeq C \begin{cases} \text{tol}^{-2} & \text{for } \beta > d\gamma \\ \text{tol}^{-2}\log^2(\text{tol}) & \text{for } \beta = d\gamma \\ \text{tol}^{-2-(d\gamma-\beta)/\alpha} & \text{for } \beta < d\gamma \end{cases}$$

Note that standard MC has corresponding complexity

$$W_{MC} \propto \text{tol}^{-2-d\gamma/\alpha}.$$

Hence,

Using a MLMC method, the MC complexity is always improved for optimal choice of N_ℓ, $\ell = 0, \ldots, L$. For $\beta = 2\alpha$, we get either $O(\text{tol}^{-2})$ (up to log terms) or $O(\text{tol}^{-d\gamma/\alpha})$.

To achieve improved complexity, one needs to:

- estimate error decay $|\mathbb{E}\left[P - P_\ell\right]|$, which is needed to determine optimal L, and

- estimate variance decay V_ℓ, which is needed to determine optimal $N_\ell{}_{\ell=0}^{L}$.

In particular,

- $|\mathbb{E}[P - P_\ell]|$ can be estimated as $|\hat{\mu}_\ell^{MC} - \hat{\mu}_{\ell-1}^{MC}|$ based on a pilot run;
- V_ℓ can be estimated by sample variance estimator based on pilot runs.

We highlight that on the finest levels we should run only very few simulations; note also that the cost for estimation of V_L might dominate the overall cost of the MLMC algorithm. To overcome this fact, we might use adaptive algorithms, in order to extrapolate information from previous levels and correct it when samples become available.

Remark 8.4.2. *It is very important to emphasize the fact that MLMC does not require the use of a geometric sequence of grids. In many applications, and most of the existing literature, a geometric sequence will be the appropriate choice (see, e.g., [30]) but there are other applications for which it is not. See, e.g., [48] and [32, 33].*

8.4.3 Application to PDEs

We consider two examples from the review paper on MLMC methods by Giles [18].

Example 1.
The first is a one-dimensional elliptic PDE, with random coefficients and random forcing. The equation is

$$\begin{cases} \dfrac{d}{dx}\left(-c(x)\dfrac{du}{dx}\right) = 50Z & x \in (0,1) \\ u(0) = u(1) = 0 \end{cases}$$

where Z is a normal random variable with zero mean and unit variance, and

$$c(x) = 1 + ax$$

with $a \sim U(0,1)$. The output quantity of interest is chosen to be

$$P = \int_0^1 u(x)dx.$$

The multilevel implementation is very easy. Level ℓ uses a uniform grid with spacing $h_\ell = 2^{-(\ell+1)}$, so there is just one interior grid point on the coarsest level. A simple second-order central difference approximation is used; the uniform second-order accuracy means that there is a constant K such that $|P - P_\ell| < Kh_\ell^2$, and therefore we have $\alpha = 2$, $\beta = 4$ and $\gamma = 1$, resulting in an $O(\text{tol}^{-2})$ complexity. All of these features can be verified in the numerical results in Figure 8.1.

Example 2.
The second example is a one-dimensional parabolic (stochastic) PDE, in which $u(x,t)$ satisfies the SPDE

$$du = \frac{\partial^2 u}{\partial x^2}dt + 10dW,$$

on the domain $0 < x < 1$ with boundary data $u(0,t) = u(1,t) = 0$ and initial data $u(x,0) = 0$; here dW denotes a Brownian motion. The output functional is chosen to be

$$P = \int_0^1 u^2(x, 0.25).$$

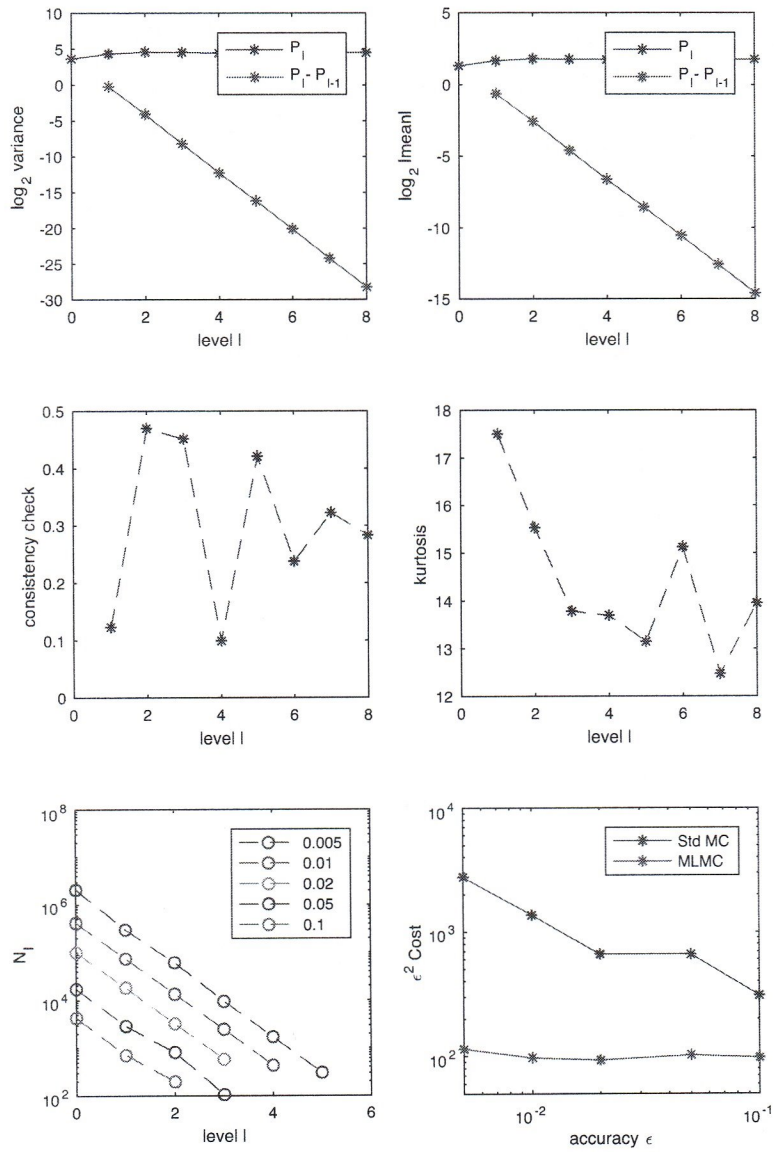


Figure 8.1: Numerical results for a one-dimensional elliptic PDE with random coefficients and random forcing. Reproduced using Matlab code taken from [18].

The multilevel implementation is again very easy. An Euler-Maruyama time discretization with time-step k, combined with a second-order space discretization with uniform grid spacing h, gives the discrete approximation

$$u_j^{n+1} = u_j^n + \frac{k}{h^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) + 10\Delta W^n.$$

The level ℓ approximation uses

$$h_\ell = 2^{-(\ell+1)}, \qquad k_\ell = h_\ell^2/4.$$

Keeping $k_\ell/h_\ell^2 = 1/4$ ensures the explicit numerical discretization is stable on all levels. Since the number of grid points doubles on each level, and the number of time-steps increases by factor 4, the cost per sample increases by factor 8, giving $\gamma = 3$. The time discretization errors are $O(k)$, and this is of the same order as the spatial discretization errors, which are $O(h^2)$, and therefore the solution error is $O(2^{-2\ell})$ and hence $\alpha = 2$ and $\beta = 4$, resulting in an $O(\text{tol}^{-2})$ complexity. These features are again confirmed in the numerical results shown in Figure 8.2.

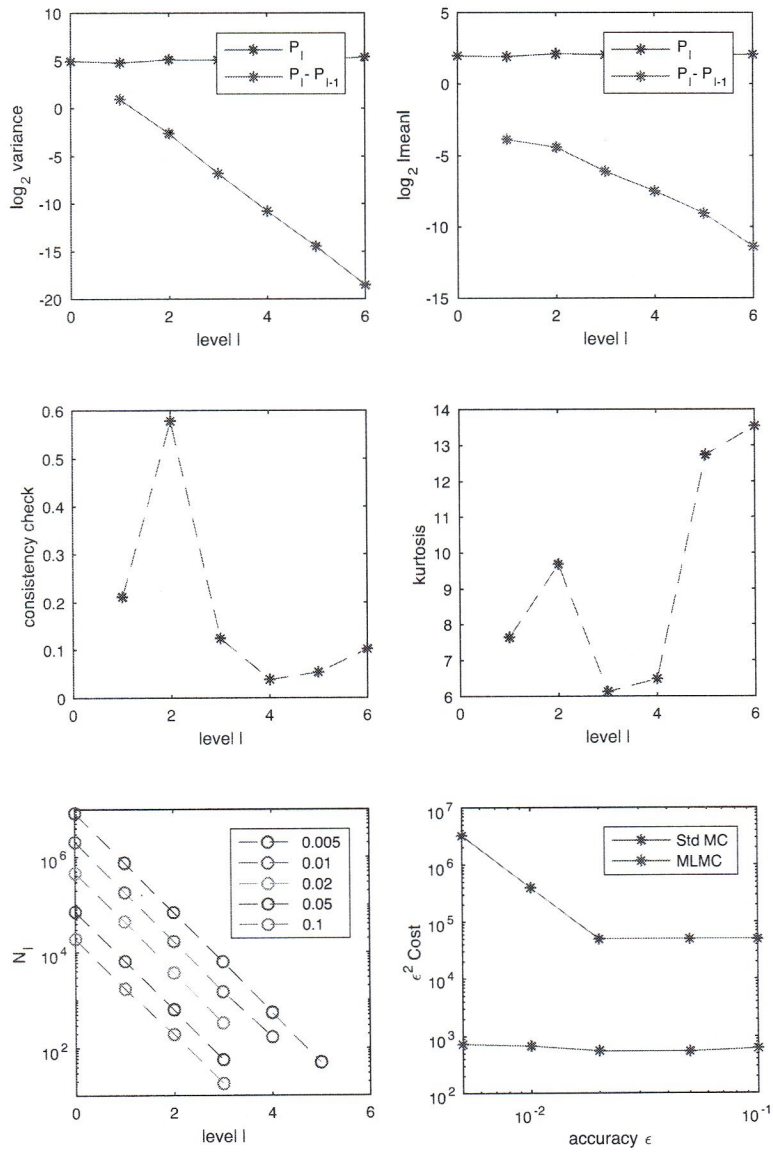


Figure 8.2: Numerical results for a one-dimensional parabolic SPDE with Brownian forcing. Reproduced using Matlab code taken from [18].

8.5 Solving PDEs with random inputs: Stochastic Finite Elements*

Another task of forward UQ is to obtain a numerical approximation of the solution of a system described in terms of PDEs or ODEs in presence of random inputs extending classical techniques – such as the finite element method. Stochastic finite element methods refer to an extensive class of algorithms for the approximate solution of partial differential equations having random input data, for which spatial discretization is effected by a finite element method. Fully discrete approximations require further discretization with respect to solution dependences on the random variables. For this purpose several approaches have been developed, including intrusive approaches such as stochastic Galerkin methods, for which the physical and probabilistic degrees of freedom are coupled, and non-intrusive approaches such as stochastic sampling and interpolatory-type stochastic collocation methods, for which the physical and probabilistic degrees of freedom are uncoupled. Following [19, 51], we sketch the construction of these techniques, pointing out the most remarkable features and computational bottlenecks.

Let us denote by $u = u(\mathbf{x}, t, \mathbf{y})$ the solution of a given PDE – omitting when necessary, hereon, the dependence on $\mathbf{x}, t$ for the sake of simplicity. Let us now suppose to be interested in the numerical approximation of the solution to a given PDE, rather than to the approximation of a given output quantity of interest $P = Q(u(\mathbf{y}))$.

8.5.1 Galerkin methods

We assume to rely on Galerkin methods for treating the spatial discretization of PDEs. *Galerkin methods* is perhaps one bit of PDE terminology that needs explaining.

Let us consider the *model* problem (8.2) already introduced in Section 8.2, taking for simplicity homogeneous Dirichlet boundary conditions:

$$\begin{cases} -\text{div}(k(\mathbf{x};\mathbf{y})\nabla u(\mathbf{x};\mathbf{y})) = & f(\mathbf{x};\mathbf{y}) & \text{in } D \\ u(\mathbf{x};\mathbf{y}) = & 0 & \text{on } \partial D, \end{cases}$$

where $\mathbf{y} = (y_1, \ldots, y_P)$ is a random vector with independent components and density $\rho(\mathbf{y})$ describing our random input – here, the diffusion coefficient or field. This might result from either a finite number of random numbers which we collect in a random vector $\mathbf{y} \in \mathbb{R}^P$, or the approximation to a random field described by a finite number of random parameters which we again collect in a random vector $\mathbf{y} \in \mathbb{R}^P$.

Let us multiply the PDE by a function v, integrate the result over D, apply Gauss' theorem to move one derivative onto v and apply the boundary condition. A Galerkin method for the PDE is defined by choosing an appropriate function space X and then seeking a function $u \in X$ that satisfies the **weak formulation** of the PDE problem:

Find $u \in V = H_0^1(D)$ such that

$$\int_D k\nabla u \cdot \nabla v d\mathbf{x} = \int_D f v d\mathbf{x} \qquad \forall v \in X.$$

A Galerkin approximation u_h of u is determined by choosing a finite dimensional subspace $X_h \subset X$, with $dim(X_h) = N_h$, and then seeking $u_h \in X_h$ that satisfies

$$\int_D k\nabla u_h \cdot \nabla v_h d\mathbf{x} = \int_D f v_h d\mathbf{x} \qquad \forall v_h \in X_h.$$

In the case of spectral Galerkin methods, the basis for X_h consists of orthogonal polynomials; in the case of finite element methods, the basis for X_h consists of piecewise polynomials defined with

respect to a grid – that is, functions that are polynomials in each grid cell and that satisfy suitable continuity requirements across cell faces.

8.5.2 Discretization with respect to stochastic parameters

The key aspect when approximating PDE with random inputs is the discretization with respect to the dependence on the parameters. *Stochastic finite element methods* (SFEMs) refer to methods for which spatial discretization is effected using finite element methods. Splitting this class of methods into subgroups, we have that:

- *Stochastic Galerkin methods* (SGMs) are a particular class of SFEMs for which discretization with respect to the random parameters is done using a Galerkin method. Polynomial chaos methods are a particular type of SGMs, for instance.

- *Stochastic sampling methods* (SSMs) are another class of SFEMs for which points in the parameter domain Γ are sampled, then used as inputs for the PDE, and then ensemble averages of outputs of interest are computed. Monte Carlo (MC) methods provide the most straightforward example of SSMs; other sampling methods include, e.g., many of the techniques already introduced, such as quasi-Monte Carlo, Latin hypercube, importance sampling (and many others). *Stochastic collocation methods* (SCMs) are also SSMs.

Hence, when approximating forward UQ problems involving PDEs, we have to discretize with respect to spatial/temporal variables and also with respect to random variables:

- for SSMs, we have to solve the discretized PDE for many parameter sample points,

- for SGMs, we have to solve a huge discrete system whose size is the product of the stochastic and finite element degrees of freedom.

Essentially, two (classes of) strategies have been developed to tackle the double curse of dimensionality (due to the potentially huge dimension of both the parameter approximation space, and the spatial approximation space), in order to do something about reducing.

- the number of stochastic degrees of freedom – e.g., develop better SSMs (e.g., *sparse grids*) that result in the same accuracy but require a lot fewer PDE solves;

- the number of spatial degrees of freedoms (examples are reduced basis methods, proper orthogonal decomposition (POD), interpolants, ...).

8.5.3 Stochastic Galerkin methods

In the Galerkin framework, one projects weighted residuals onto a finite-dimensional subspace spanned by appropriate basis functions to provide the constraints required to solve for the deterministic coefficients. For collocation methods, the constraints are provided by approximating the solution of the governing equations at a discrete set of points (collocation points) in the random variable space used to represent inputs.

Stochastic Galerkin methods (SGMs) are stochastic finite element methods for which discretization with respect to parameter space is also effected using a Galerkin approach. In this framework, it is natural to then treat the solution $u(\mathbf{x};\mathbf{y}) \in X \times Z$ of the PDE with random inputs as function of $d+P$ variables, i.e., of the d spatial variables plus the P random parameters – here we usually consider the space Z as given by $Z = L^2_\rho(\Gamma)$. Hence, the solution fulfills the following **stochastic weak formulation**:

Find $u(x;\mathbf{y}) \in X \otimes Z$ such that

$$\int_\Gamma \int_D k(\mathbf{x};\mathbf{y}) \nabla u(\mathbf{x};\mathbf{y}) \cdot \nabla v(\mathbf{x};\mathbf{y}) d\mathbf{x} \rho(\mathbf{y}) d\mathbf{y} = \int_\Gamma \int_D f((\mathbf{x};\mathbf{y}) v(\mathbf{x};\mathbf{y}) d\mathbf{x} \rho(\mathbf{y}) d\mathbf{y} \qquad \forall v \in X \otimes Z.$$

Equivalently,

Find $u \in L^2_\rho(\Gamma; H^1_0(D)) \sim L^2_\rho(\Gamma) \otimes H^1_0(D)$ is such that

$$\mathbb{E}\left[\int_D k(\mathbf{x};\mathbf{y}) \nabla u \cdot \nabla v d\mathbf{x}\right] = \mathbb{E}\left[\int_D f(\mathbf{y}, x) v(\mathbf{y}, x) d\mathbf{x}\right] \qquad \forall v \in L^2_\rho(\Gamma) \otimes H^1_0(D).$$

The finite element (FE) approximation in the physical space is performed by introducing a suitable FE space $H_h(D) \subset H^1_0(D)$ and a basis $\{\phi_i(\mathbf{x})\}_{i=1}^{N_h}$, and by seeking a solution of the form

$$u_h(\mathbf{x};\mathbf{y}) = \sum_{i=1}^{N_h} u_i(\mathbf{y}) \phi_i(\mathbf{x}),$$

due to the nature of the domain $D \times \Gamma$ and space $L^2_\rho(\Gamma) \otimes H^1_0(D)$. Note that each dof (nodal value) $u_i = u_i(\mathbf{y})$ is a random variable. The **semi-discrete finite element approximation** reads as follows:

Find $u_h \in L^2_\rho(\Gamma) \otimes H_h(D)$ is such that

$$\mathbb{E}\left[\int_D k(\mathbf{x};\mathbf{y}) \nabla u_h(\mathbf{x};\mathbf{y}) \cdot \nabla v_h(\mathbf{x};\mathbf{y}) d\mathbf{x}\right] = \mathbb{E}\left[\int_D f(\mathbf{x};\mathbf{y}) v_h(\mathbf{x};\mathbf{y}) d\mathbf{x}\right] \qquad \forall v_h \in L^2_\rho(\Gamma) \otimes H_h(D).$$

If we discretize in parameter space using a basis consisting of tensor products of global orthogonal (with respect to the pdf ρ) polynomials, we get what is commonly referred to as *polynomial chaos approximations*.

Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_P) \in \mathbb{N}^P$ be a multi-index, with $\alpha_i \in \mathbb{N}$, and let $\Lambda(w) \subset \mathbb{N}^P$ be an index set of cardinality M_w. A possible choice[6] is

$$\Lambda(w) = \{\boldsymbol{\alpha} \in \mathbb{N}^P \ : \max_{\boldsymbol{\alpha} \in \Lambda} \max_{n=1,\ldots,P} \alpha_n = w\}$$

so that the maximum polynomial degree in each variable y_n is at most w. Then, we consider the multivariate polynomial space

$$\mathbb{P}_{\Lambda(w)}(\Gamma) = \text{span}\left\{\prod_{n=1}^{P} y_n^{\alpha_n}, \ \text{ with } \boldsymbol{\alpha} \in \Lambda(w)\right\}$$

and let $\{\Psi_i\}_{i=1}^{M_w}$ a basis of $\mathbb{P}_{\Lambda(w)}(\Gamma)$ (e.g., multivariate Legendre, Hermite, ... orthogonal polynomials). We look for a full discrete solution $u_{h,w} \in \mathbb{P}_{\Lambda(w)}(\Gamma) \otimes H_h(D)$ of the form

$$u_{h,w} = \sum_{j=1}^{N_h} \sum_{i=1}^{M_w} u_{ij} \Psi_i(\mathbf{y}) \phi_j(x).$$

One way to determine the approximation $u_{h,w} \in \mathbb{P}_{\Lambda(w)}(\Gamma) \otimes H_h(D)$ is by Galerkin projection, yielding the following **fully discrete finite element approximation**:

[6] In Chapter 7 we encountered the case of $\Lambda(w) = \{\boldsymbol{\alpha} \in \mathbb{N}^P \ : \ |\boldsymbol{\alpha}| \leq w\}$.

Find $u_{h,w}^{SG} \in \mathbb{P}_{\Lambda(w)}(\Gamma) \otimes H_h(D)$ is such that

$$\mathbb{E}\left[\int_D k(\mathbf{x};\mathbf{y})\nabla u_{h,w}^{SG}(\mathbf{x};\mathbf{y}) \cdot \nabla v_h(\mathbf{x};\mathbf{y})dx\right] = \mathbb{E}\left[\int_D f(\mathbf{x};\mathbf{y})v_h(\mathbf{x};\mathbf{y})dx\right] \quad \forall v_h \in L^2_\rho(\Gamma)\otimes H_h(D).$$

Expanding on the basis $\{\phi_i(x)\} \times \{\Psi_j(\mathbf{y})\}$ and using orthogonal polynomials $\{\Psi_j\}$ with respect to $\rho(\mathbf{y})$ yields

$$\sum_{l=1}^{M_w}\sum_{k=1}^{N_h} u_{lk} \int_D \mathbb{E}\left[k(\mathbf{x};\mathbf{y})\Psi_l(\mathbf{y})\Psi_i(\mathbf{y})\right]\nabla\phi_k(\mathbf{x}) \cdot \nabla\phi_j(\mathbf{x})d\mathbf{x} = \int_D \mathbb{E}\left[f(\mathbf{x};\mathbf{y})\Psi_i(\mathbf{y})\right]\phi_j(\mathbf{x})d\mathbf{x}$$

for all $j = 1,\dots,N_h$, $i = 1,\dots,M_w$. Then, numerical quadrature formulas must be used to approximate the integrals in the P-dimensional parameter space, as well as the spatial integrals.

The discrete stochastic Galerkin system one has to solve is huge, and involves $N_h \times M_w$ equations and unknowns u_{lk}, where M_w is the number of parameter degrees of freedom, and N_h is the number of finite element degrees of freedom. Note that we need to compute, at every quadrature point $\mathbf{x}_s$ in the physical space, a term of the form

$$c_{il}(\mathbf{x}_s) = \mathbb{E}\left[k(\mathbf{x}_s;\cdot)\Psi_i\Psi_l\right] = \int_\Gamma k(\mathbf{x}_s;\mathbf{y})\Psi_i(\mathbf{y})\Psi_l(\mathbf{y})\rho(\mathbf{y})d\mathbf{y},$$

a task that can easily become challenging.

8.5.4 Stochastic collocation methods

Stochastic Galerkin methods are mathematically very attractive in that they are a natural extension of the Galerkin methods that are commonly used for deterministic PDEs, but has the severe disadvantage that the stochastic modes of the solution are coupled together by a large system.

In contrast, non-intrusive spectral methods for UQ are characterized by the feature that the solution of the deterministic problem is a 'black box' that does not need to be modified for use in the spectral method, beyond being able to be evaluated at any desired point $\mathbf{y}$ of the probability space.

Stochastic collocation methods (SCMs) are non-intrusive, sampling-based methods. The term "*collocation*" originates from the deterministic numerical methods for differential equations, where one seeks to satisfy the governing continuous equations discretely at a set of collocation points[7] (think for instance to finite difference methods...). The same definition can be extended to stochastic simulations. In their most general sense, SCMs share the same philosophy of stochastic sampling methods, such as Monte Carlo methods; their goal is, however, to construct an accurate approximation to the problem solution, rather than estimating solution's statistics or output quantities of interest depending on the problem solution.

To illustrate the idea, let us consider again the model problem (8.2) set on a spatial domain D:

$$\begin{cases} -\text{div}(k(\mathbf{x};\mathbf{y})\nabla u(\mathbf{x};\mathbf{y})) = f(\mathbf{x};\mathbf{y}) & \text{in } D \\ u(\mathbf{x};\mathbf{y}) = 0 & \text{on } \partial D, \end{cases} \tag{8.7}$$

where we recall that $\Gamma \subset \mathbb{R}^P$, $P \geq 1$ is the support of the uncertain parameters $\mathbf{y} = (y_1,\dots,y_P)$; the solution of the problem is then a mapping $u(\mathbf{x};\mathbf{y}) : D \times \Gamma \to \mathbb{R}$. For ease of notation, hereon we will only denote the dependence of u on $\mathbf{y}$

Let $\Theta_M = \{\mathbf{y}^{(j)}\}_{j=1}^M \subset \Gamma$ be a set of (prescribed) nodes (or *collocation points*) in the random space, where $M \geq 1$ is the number of nodes. Then in SC, we enforce (8.7) at the node $\mathbf{y}^{(j)}$ for all

[7] This is to the contrary of Galerkin method, where one seeks to satisfy the governing equation in a weak form.

$j = 1, \ldots, M$ by solving

$$\begin{cases} -\text{div}(k(\mathbf{x};\mathbf{y}^{(j)})\nabla u(\mathbf{y}^{(j)})) = & f(\mathbf{x};\mathbf{y}^{(j)}) & \text{in } D \\ u(\mathbf{y}^{(j)}) = & 0 & \text{on } \partial D, \end{cases} \tag{8.8}$$

It is easy to see that for each j, (8.8) is a deterministic problem because the value of the random parameter $\mathbf{y}$ is fixed. Therefore, solving the system poses no difficulty provided one has a well-established deterministic algorithm.

Let $u^{(j)} = u(\mathbf{y}^{(j)})$, $j = 1, \ldots, M$ be the solution of the above problem. The result of solving (8.8) is an ensemble of deterministic solutions $\{u^{(j)}\}_{j=1}^{M}$, and one can apply various post-processing operations to the ensemble to extract useful information about $u(\mathbf{y})$.

The goal of SC is to construct a numerical approximation $u^{SC}(\mathbf{y}) : \Gamma \to \mathbb{R}$ in a proper polynomial space to the solution response $u(\mathbf{y}) : \Gamma \to \mathbb{R}$ using the deterministic solution ensemble $\{u(\mathbf{y}^{(j)})\}$, $j = 1, \ldots, M$, of (8.8), in the sense that $\|u^{SC}(\mathbf{y}) - u(\mathbf{y})\|$ is sufficiently small in a strong norm defined on Γ – typically, an L^p-norm ($p \geq 1$), with the L^2-norm used the most in practice and leading to "mean-square" approximation.

The numerical approximation $u^{SC}(\mathbf{y})$ shall be chosen from a class of functions; the most widely used choice is a polynomial space, which leads to strong ties of SC methods to *generalized polynomial chaos* (gPC) approximation. The construction and properties of SC methods then critically depend on the approximation properties of u_{SC} and the choice of the collocation nodal set Θ_M . The most common choices are SC methods of *interpolation* type or *projection* type. Here we focus on the former, since interpolation is a natural approach to the stochastic collocation problem.

Given the nodal set $\Theta_M = \{\mathbf{y}^{(j)}\}_{j=1}^{M}$ and the M deterministic solutions $\{u(\mathbf{x};\mathbf{y}^{(j)})\}_{j=1}^{M}$, find a polynomial $u^{SC}(\mathbf{x};\mathbf{y})$ in a suitable polynomial space Π such that

$$u^{SC}(\mathbf{x};\mathbf{y}^{(j)}) = u(\mathbf{x};\mathbf{y}^{(j)}) \qquad \forall j = 1, \ldots, M.$$

The goal can be easily accomplished, at least in principle. One way is to use a *Lagrange interpolation approach*[8]. That is, we seek

$$u^{SC}(\mathbf{x};\mathbf{y}) = \sum_{j=1}^{M} u(\mathbf{x};\mathbf{y}^{(j)}) L_j(\mathbf{y})$$

where

$$L_j(\mathbf{y}^{(i)}) = \delta_{ij}, \qquad i, j = 1, \ldots, M$$

are the Lagrange interpolating polynomials. By construction, the polynomial $u^{SC}(\mathbf{y})$ automatically satisfies the interpolation conditions. We then need to explicitly construct the Lagrange interpolation polynomials $L_j(\mathbf{y})$ – a straightforward task in dimension $P = 1$, less trivial in multiple dimension $P > 1$.

The other way is a *matrix inversion approach*, where we prescribe the polynomial interpolating basis first and a Vandermonde-like system has to be solved to fulfill the interpolation constraints.

More specifically, let $u^{SC} \in V_N$ be constructed from a linear space V_N with cardinality $dim(V_N) = N$. Let $(b_1, \ldots, b_N)$ be a basis for V_N . Then, we can express u^{SC} as

$$u^{SC}(\mathbf{x};\mathbf{y}) = \sum_{i=1}^{N} c_i(\mathbf{x}) b_i(\mathbf{y}) \tag{8.9}$$

[8] We all know that to construct robust and accurate interpolations, one should employ grids that are clustered toward the boundaries of the interval. The well-known example of interpolating the Runge function clearly illustrates this property.

where $c_i(\mathbf{x})$ are coefficient functions to be determined. We then enforce the interpolation condition

$$u^{SC}(\mathbf{x};\mathbf{y}^{(j)}) := \sum_{i=1}^{N} c_i(\mathbf{x}) b_i(\mathbf{y}^{(j)}) = u(\mathbf{x};\mathbf{y}^{(j)}) \qquad \forall j = 1,\dots,M.$$

Thus, each of the coefficient functions $\{c_i(\mathbf{x})\}_{i=1}^N$ is a linear combination of the finite element data $\{u(\mathbf{x};\mathbf{y}^{(j)})\}_{j=1}^M$; the specific linear combinations are determined in the usual manner from the entries of the inverse of the interpolation matrix A, where

$$(A)_{ij} = b_j(\mathbf{y}^{(i)}), \qquad i = 1,\dots,M, \quad j = 1,\dots,N.$$

Remark 8.5.1. *The main attraction of interpolatory approximations of parameter dependences is that it effects a complete decoupling of the spatial and probabilistic degrees of freedom. The solution of the deterministic finite element problems, one for each parameter point $\mathbf{y}^m$, disregards which basis $\{b_i(\mathbf{y})\}_{j=1}^N$ we choose to use. It is only in the definition of the Vandermonde matrix that the choice of stochastic basis enters into the picture.*

For example, if one adopts the gPC expansion, then

$$u_N^{SC}(\mathbf{y}) = \sum_{|\mathbf{k}|=0}^{N} \hat{u}_{\mathbf{k}}^{SC} \Phi_{\mathbf{k}}(\mathbf{y})$$

where the P-variate Nth-degree gPC basis functions are the products of the univariate gPC polynomials of total degree less than or equal to N,

$$\Phi_{\mathbf{i}}(\mathbf{y}) = \phi_{i_1}(y_1) \times \cdots \times \phi_{i_P}(y_P), \qquad 0 \le |\mathbf{i}| \le N,$$

and are orthogonal polynomials chosen based on the probability distribution of $\mathbf{y}$, the Vandermonde matrix will be given by

$$(A)_{ij} = \Phi_{\mathbf{j}}(\mathbf{y}^{(i)}), \qquad i = 1,\dots,M, \quad 0 \le |\mathbf{j}| \le N.$$

The advantage of the matrix inversion approach is that the interpolating polynomials are prescribed and well defined. However, an important and very practical concern is the accuracy of the interpolation. Even though the interpolation has no error at the nodal points, error can become wild between the nodes, and this is particularly true in high-dimensional spaces. To prevent the problem from becoming underdetermined, we require the number of collocation points not to be smaller than the number of gPC expansion terms, i.e., $M \ge \binom{N+P}{N}$. When the number of the collocation points is the same as the number of the basis functions, i.e., $M = \binom{N+P}{N}$, the matrix A is square and can be inverted when it is nonsingular. Although very flexible, this approach is not used widely in practice. For this reason, a least-square approach is often preferred.

Remark 8.5.2. *In any case, the big issue in this context is how to perform interpolation in multiple dimensions ($P > 1$). Extending the well-studied one-dimensional interpolation methods to multiple dimensions via tensor product rules results in sampling sets that are structured, but the growth of the number of samples in high dimensions can be prohibitively fast ("curse of dimensionality"). For this reason, it is usually preferred to construct interpolations on a set of unstructured nodes, e.g. through so-called (Smolyak) sparse grids interpolations, which are built through judicious sparsifications of tensor product rules (see Figure 8.3), and also allow* anisotropic *choices of interpolation points to take into account the importance of different parameters. But this would deserve another course...*

We only highlight that to obtain good results when using sparse grids, and to get away with such big empty spaces in the sampling pattern, we need to ensure that the solution $u(\mathbf{x};\mathbf{y})$ is smooth with respect to the parameters $\mathbf{y}$. For an integrand that is not sufficiently smooth, the sparse-grid approach using global polynomial interpolatory quadrature rules result in generally very bad approximations – the same comment holds for global orthogonal polynomial approximations, i.e., for polynomial chaos methods.

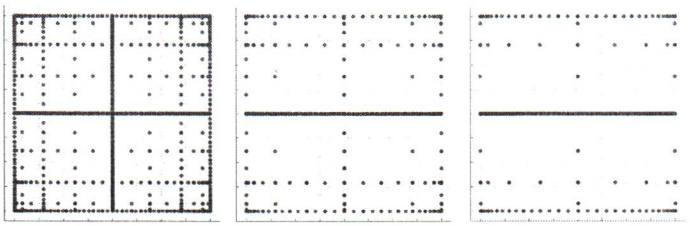


Figure 8.3: Sparse grids subsets of tensor product grids. Left: isotropic sparse grid. Center and right: anisotropic sparse grids that take into account the importance of different parameters.

Remark 8.5.3. *An alternative option to the interpolation approach would be a (discrete) projection approach, a* discrete *gPC projection is performed. Recall from Section 7.7.1 that a gPC approximation of $u(\mathbf{y})$ can be obtained from the exact orthogonal gPC projection of $u(\mathbf{y})$,*

$$u_N(\mathbf{y}) = P_N u(\mathbf{y}) = \sum_{\mathbf{k}=0}^{N} \hat{u}_{\mathbf{k}} \Phi_{\mathbf{k}}(\mathbf{y}),$$

where the $\mathbf{k}$-th expansion (or Fourier) coefficient of u is

$$\hat{u}_{\mathbf{k}} = \frac{1}{a_{\mathbf{k}}} \mathbb{E}\left[u(\mathbf{y})\Phi_{\mathbf{k}}(\mathbf{y})\right] = \frac{1}{a_{\mathbf{k}}} \int u(\mathbf{y})\Phi_{\mathbf{k}}(\mathbf{y}) f_y(\mathbf{y}) d\mathbf{y}, \qquad |\mathbf{k}| \leq N$$

where $a_{\mathbf{k}}$ are the normalization constants of the basis. The idea of discrete projection is to approximate the integrals in the expansion coefficients above, of the continuous generalized polynomial chaos (gPC) projection $u_N(\mathbf{y})$ by an integration rule, thus yielding

$$u_N^{PC}(\mathbf{y}) = \sum_{|\mathbf{k}|=0}^{N} \tilde{u}_{\mathbf{k}} \Phi_{\mathbf{k}}(\mathbf{y}).$$

Once again, numerical integration and interpolation are key tools...

Remark 8.5.4. *What to choose, then, between stochastic Galerkin and stochastic collocation methods? The stochastic Galerkin approach ensures that the residual of the stochastic governing equations is orthogonal to the linear space spanned by the gPC polynomials. In this sense, the accuracy is optimal. On the other hand, stochastic collocation approaches, with no error at the nodes, introduce errors either because of the interpolation scheme (if interpolation collocation is used) or because of the integration rule (if discrete projection collocation is used). Both errors are caused by the introduction of the nodal sets. Though in one dimension, these errors can be kept at the same order as the error of the Galerkin method, in multidimensional spaces these errors can be much more significant, unless suitable choices of integration rules or interpolation points are selected.*

8.5.5 Stochastic collocation for a heat transfer problem

For the sake of showing nice pictures, we close this (tonic) chapter propose an example of (adaptive, anisotropic) stochastic collocation method for a heat transfer problem, modeled through the unsteady heat equation for the temperature $T = T(\mathbf{x}, t; \omega)$

$$\begin{cases} \partial_t T - \nabla \cdot (\alpha(\mathbf{x},\omega)\nabla T) &= 0 \quad \text{in } D \times (0,1] \times \Omega \\ -\alpha(\mathbf{x},\omega)\ \nabla T \cdot \mathbf{n} &= 0 \quad \text{on } \partial D_1 \times (0,1] \times \Omega \\ -\alpha(\mathbf{x},\omega)\ \nabla T \cdot \mathbf{n} &= -1 \quad \text{on } \partial D_2 \times (0,1] \times \Omega \\ T(\mathbf{x},0,\omega) &= 0 \quad \text{in } D \times \Omega \end{cases} \tag{8.10}$$

The problem describes the heat conduction in an electronic chip with the uncertain heat conductivity α, modeled as a random field, discretized by means of a truncated Karhunen-Loeve expansion with 3 uniformly distributed random variables; as a consequence, the random input is $\mathbf{y} = (y_1, y_2, y_3)$. Figure 8.4 shows the spatial domain D. The time interval is $(0, 1]$.

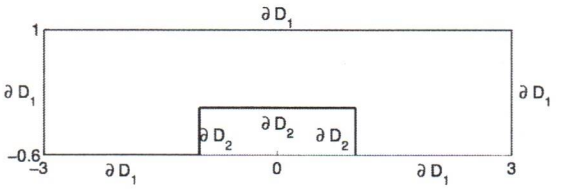


Figure 8.4: Spatial domain for the solution of the heat equation depending on random inputs.

Check `https://nbviewer.jupyter.org/github/bschieche/MATLAB-random-PDE/blob/master/stochastic-collocation-heat-equation.ipynb` for further details and the `Matlab` code.

The conductivity parameter α is the source of uncertainty, and is modeled as a spatially correlated random field with expected value $\mathbb{E}[\alpha] = 1$, correlation length $L_c = 4$ and standard deviation $\sigma = 0.2$, using a Matérn covariance function

$$C_\alpha(\mathbf{x}, \tilde{\mathbf{x}}) = \sigma^2 \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|_2}{L_c} K_1\left(\frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|_2}{L_c}\right), \tag{8.11}$$

We evaluate the covariance spectrum numerically on the same spatial discretization used for the FE calculations (linear finite elements). The first three modes are reported in Figure 8.5.

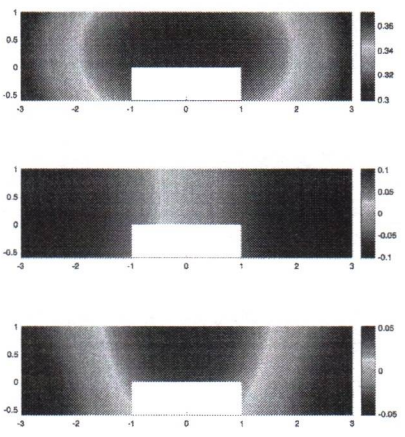


Figure 8.5: The first three eigenfunctions of the covariance function

The discretization in space and time is done with triangular finite elements and the `Matlab` `ode15s` solver for stiff problems, respectively. The random part is discretized by means of an interpolation with adaptive, anisotropic placement in the nodes of a sparse grid. To minimize the interpolating nodes is especially important, because each node means solving a deterministic PDE, which is computationally demanding in many cases. In this respect, *sparse grids* provide a very effective tool, provided the dimension of the random inputs space is not too large.

The generalized Smolyak algorithm is used with global hierarchical Lagrange polynomials as basis functions; Gauss-Patterson Legendre collocation points are then considered. Each collocation point then means running a deterministic PDE (discretized in space and time). For the presentation of the results, six reference points in the spatial domain D are chosen, namely $(0,0)$, $(0,0.5)$, $(1,0.5)$, $(2,0.5)$, $(2,0)$, and $(1,-0.6)$. .

We also plot the mean of the solution and its variance, see Figure 8.6. In all plots we observe that there is more variance in the solution near the cavity. Moreover, the time evolution reveals that the error bars seem to run into steady states. While the solution keeps growing, this is apparently not true for the variance.

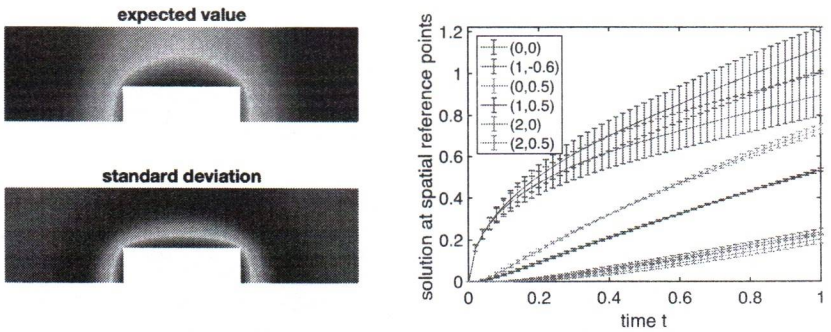


Figure 8.6: Left: mean and variance of the PDE solution at the final time. Right: time evolution of the solution at some selected points in the spatial domain.

Figure 8.7 shows the full functional dependency of the output Quantity of interest (which is the time integral over the most active point in the cavity) of the random variables image space projected on the first two random dimensions for illustration purposes. We see a slightly nonlinear behavior especially in the dominating, first dimension. To resolve the first dimension adequately, the algorithm places more collocation points in this direction. This example thus demonstrates nicely the anisotropic behavior of the stochastic collocation procedure.

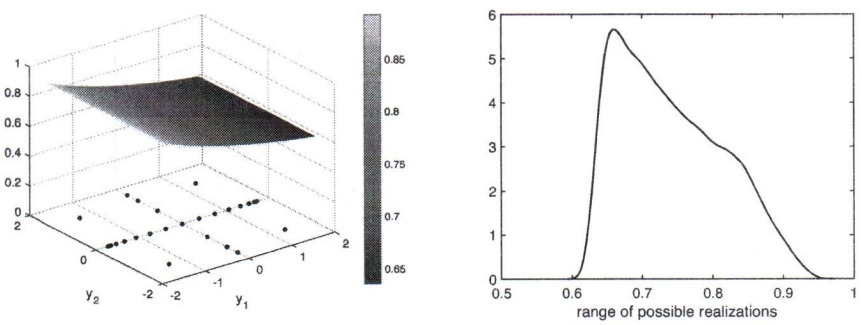


Figure 8.7: Forward UQ on a possible output of interest (the time integral over the most active point in the cavity).

We observe a quite long tail in the density of the output QoI to its right side; hence, although uniformly distributed inputs are considered, the output of the simulation is characterized by a different probability distribution, thus showing, once again, the importance of performing forward uncertainty quantification.

8.6 Further readings, code and material

The literature on forward UQ problems is growing fast in the last decade. Several papers in the last decade have focused on PDEs (and stochastic PDEs): see (Barth, Schwab and Zollinger 2011, Charrier, Scheichl and Teckentrup 2013, Cliffe et al. 2011, Teckentrup, Scheichl, Giles and Ullmann 2013) for elliptic PDEs, (Barth, Lang and Schwab 2013, Giles and Reisinger 2012) for parabolic PDEs, (Mishra, Schwab and Sukys 2012, Mishra, Schwab and Sukys 2016) for hyperbolic PDEs. In almost all of this work, the construction of the multilevel estimator is quite natural, using a geometric sequence of grids and the usual estimators for $g_\ell - g_{\ell-1}$. It is the numerical analysis of the variance of the multilevel estimator which is often very challenging. More recent developments can be found, e.g., in:

- [30], where a MLMC method with control variate has been applied to elliptic PDEs with log-normal coefficients;
- [28], [29] where MLMC methods have been applied to nonlinear systems of conservation laws or to uncertainty quantification of acoustic wave propagation in random heterogeneous layered medium;
- [16], [15] where a multifidelity control variate approach for the multilevel Monte Carlo technique has been proposed, and then applied to the simulation of high-dimensional uncertain systems.

More recently, control variates and multi fidelity Monte Carlo techniques (MFMC), in spirit similar to MLMC, have been applied to several contexts. Multi fidelity Monte Carlo estimation has been considered in [32] and has been applied to large-scale uncertainty propagation in [31] and sensitivity analysis in [38]; see also [33].

Regarding instead stochastic finite element methods, an extremely detailed bibliographic review can be found in [19]. Among well-known results about stochastic collocation methods for elliptic PDEs, we mention [5] and [6], where a comparison among stochastic Galerkin and stochastic collocation has been performed. See also, e.g., part III of [17].

Regarding basic implementations of MLMC/MFMC codes, it is possible to find some `Matlab` or `Python` packages at:

- `https://people.maths.ox.ac.uk/gilesm/mlmc.html`, a Matlab implementation of the MLMC methods described in [18];
- `https://github.com/nasa/MLMCPy`, a MLMC package developed and maintained by the UQ group at NASA;

  `https://github.com/pehersto/mfmc` and `https://github.com/elizqian/mfgsa`, implementing the methods proposed in [32], [38].

Instead, `Matlab` or `Python` packages performing stochastic Galerkin or stochastic collocation methods can be found, e.g., at:

- `https://sites.google.com/view/sparse-grids-kit`. `Sparse Grids Matlab Kit` is a collection of Matlab functions for high-dimensional quadrature and interpolation, based on sparse grids.
- `http://www.openturns.org`. `OpenTURNS` is a C++/Python software for uncertainty quantification, uncertainty propagation, sensitivity analysis and meta-modeling.
- `https://nbviewer.jupyter.org/github/bschieche/MATLAB-random-PDE/blob/master/stochastic-collocation-heat-equation.ipynb`