Efficient multilevel and multi-index methods in Uncertainty Quantification

Pieterjan Robbe
joint work with D. Nuyens and S. Vandewalle

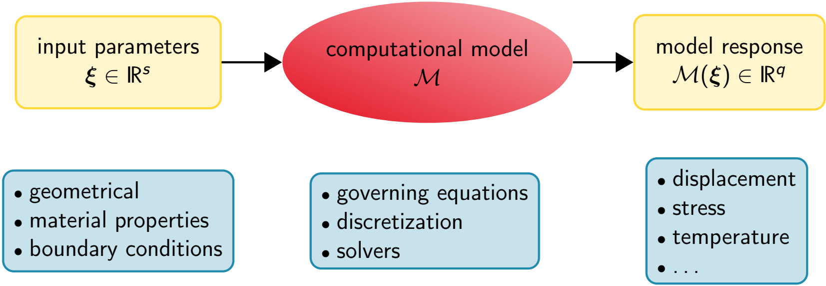
WSC Spring Meeting

KU Leuven - University of Leuven
Celestijnenlaan 200A
B3001 Leuven
Belgium
✉ pieterjan.robbe@kuleuven.be

Contents

Computational models

- Computational model as **black box**



- If input parameters or model are subject to uncertainty
  $\Rightarrow$ **uncertainty quantification** or UQ

Specific industrial challenges

What makes industrial UQ problems hard?

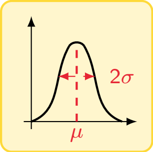
- Computational models are **complex**: nonlinearity, coupled problems (thermo-mechanics), plasticity, contact zones, ...
- Simulations are **costly**: a single run can take up to several hours or days, or more
- Number of inputs is typically 10-1000: **high-dimensional** problems (possibly even infinite-dimensional)
- UQ code comes on top of well defined simulation procedures

Engineers focus on a so-called **quantity of interest** $g = \mathcal{F}[\mathcal{M}(\boldsymbol{\xi})]$, such as maximum displacement, average stress, ...

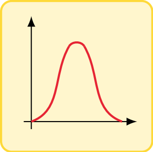
Typical engineering questions

Typical outcomes of the uncertainty propagation phase are:

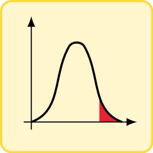
- Statistics of the quantity of interest



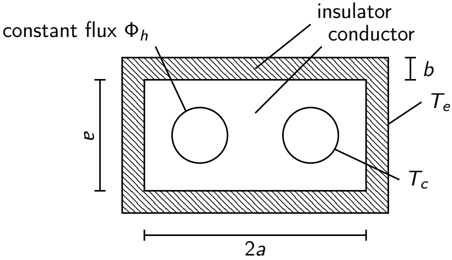
- Distribution of the quantity of interest



- Failure probability of the quantity of interest



Motivational example

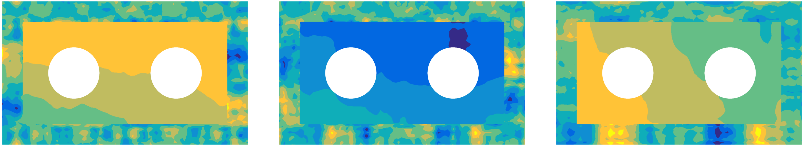


- Idealized model for a two-dimensional heat exchanger
- Conductor material k^{int} modelled with "smooth" variation
- Insulator material k^{int} modelled with "rough" variation

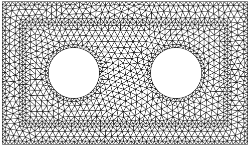
- Quantity of interest g is maximum temperature

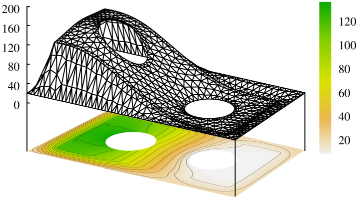
Motivational example

- Some example visualisations of the material



- Example mesh and mean temperature field





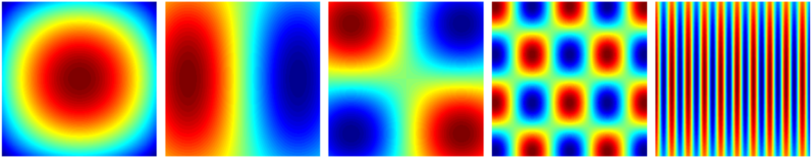
Modeling spatial variation

- Represent the conductivity as a **lognormal** random field

$$k(\boldsymbol{x},\omega) = \exp(Z(\boldsymbol{x},\omega))$$

  with $Z(\boldsymbol{x},\omega)$ a Gaussian random field
- Every sample $\omega \in \Omega$ yields a **realisation** of the random field
- Classical technique to generate realisations of $k(\boldsymbol{x},\omega)$ is the **KL-expansion**

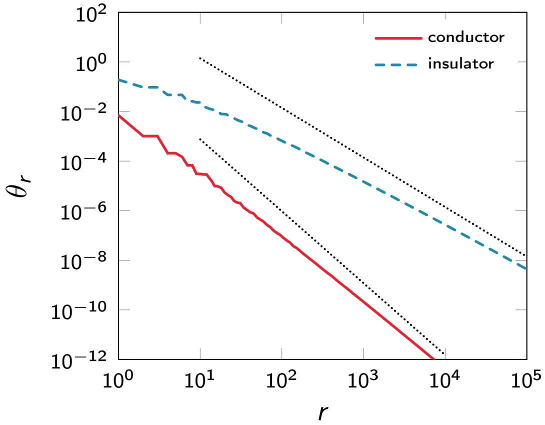
$$k(\boldsymbol{x},\omega) = \exp\left(\mu(\boldsymbol{x}) + \sum_{r=1}^{\infty} \sqrt{\theta_r} f_r(\boldsymbol{x}) \xi_r(\omega)\right)$$



$r = 1$ $\quad$ $r = 2$ $\quad$ $r = 4$ $\quad$ $r = 15$ $\quad$ $r = 88$

The KL expansion

- Approximation quality of the KL expansion determined by eigenvalue decay rate



The KL expansion

- Eigenvalues and eigenfunctions are solutions of the Fredholm equation

$$\int_D C(\boldsymbol{x}, \boldsymbol{y}) f_r(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} = \theta_r f_r(\boldsymbol{y}), \quad \boldsymbol{x}, \boldsymbol{y} \in D$$

  where $C(\boldsymbol{x}, \boldsymbol{y})$ is the **covariance function** of the random field
- Faster decay of the eigenvalues θ_r gives a more smooth random field
- In practice, the expansion must be truncated after a **finite number of terms** s
- Higher s means better approximation, but also higher cost (eigenvalue problem + evaluation)
- Algorithms that take advantage of this property?

Governing equations

- Linear anisotropic steady-state **stochastic heat equation** on a domain $D \in \mathbb{R}^d$ with $d = 2$ and boundary ∂D
- We wish to compute the temperature field $T : D \times \Omega \to \mathbb{R} : (\boldsymbol{x}, \omega) \mapsto T(\boldsymbol{x}, \omega)$ that solves almost surely

$$-\nabla \cdot \left[k(\boldsymbol{x}, \omega) \nabla T(\boldsymbol{x}, \omega) \right] = F(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \in D \text{ and } \omega \in \Omega$$

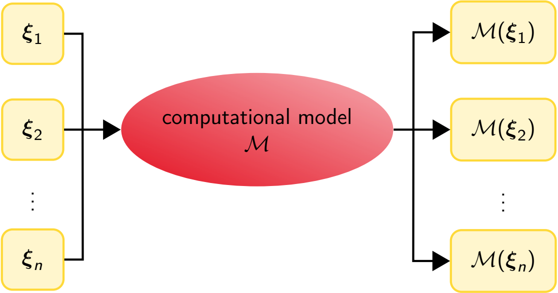
where the event ω belongs to a **probability space** $(\Omega, \mathcal{F}, P)$

- For the KL expansion of a Gaussian field, we take $\Omega = \mathbb{R}^s$
- Given (deterministic) boundary conditions

$$\begin{aligned} T(\boldsymbol{x}, \cdot) &= T_1(\boldsymbol{x}) \qquad && \text{for } \boldsymbol{x} \in \partial_1 D \\ n(\boldsymbol{x}) \cdot (k(\boldsymbol{x}, \cdot) \nabla T(\boldsymbol{x}, \cdot)) &= T_2(\boldsymbol{x}) \qquad && \text{for } \boldsymbol{x} \in \partial_2 D \end{aligned}$$

Uncertainty propagation using Monte Carlo

- A sample set $\{\xi_1, \xi_2, \ldots, \xi_n\}$ is drawn according to the input distributions f_X
- For each sample, the quantity of interest is evaluated



- The set of output quantities $\{\mathcal{M}(\xi_1), \mathcal{M}(\xi_2), \ldots, \mathcal{M}(\xi_n)\}$ is then used for analysis, for example

$$\mathbb{E}[g] \approx Q(g) := \frac{1}{n} \sum_{i=1}^{n} \mathcal{F}\left[\mathcal{M}(\xi_i)\right]$$

Advantages/drawbacks of Monte Carlo

Advantages

- **Universal**: only requires samples from an input pdf and repeated model evaluations
- **Convergence** under mild conditions: law of large numbers and central limit theorem, requires L_2 integrability
- **Parallel**: all samples are independent, hence suitable for high-perfomance computing

Drawbacks

- **Statistical uncertainty**: result is typically given with confidence interval: $Y = a \pm b$ with c% confidence
- **Low efficiency**: convergence rate is $\mathcal{O}(1/\sqrt{N})$, where N is the number of realisations

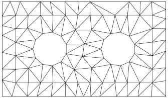
Multilevel idea

- Implicitly assumed that model is discretized
- **Multilevel idea**: suppose we have multiple discrete approximations g_ℓ available with different accuracies, called levels $\ell = 0, 1, 2, \ldots$
- **Telescoping sum**:

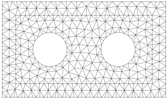
$$\mathbb{E}[g_L] = \mathbb{E}[g_0] + \sum_{\ell=1}^{L} \mathbb{E}[g_\ell - g_{\ell-1}] = \sum_{\ell=0}^{L} \mathbb{E}[\Delta g_\ell]$$

- Huge cost reduction if

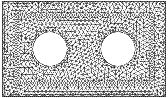
$$\mathbb{V}[\Delta g_\ell] \to 0 \ \text{ fast for } \ \ell \to \infty$$



$\ell = 0$



$\ell = 1$



$\ell = 2$

Multi-index idea

- **Extension**: assume that g is discretized to $g_{\boldsymbol{\ell}}$, where the components of $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m)$ are different discretization dimensions
- Define **difference operator** in direction i

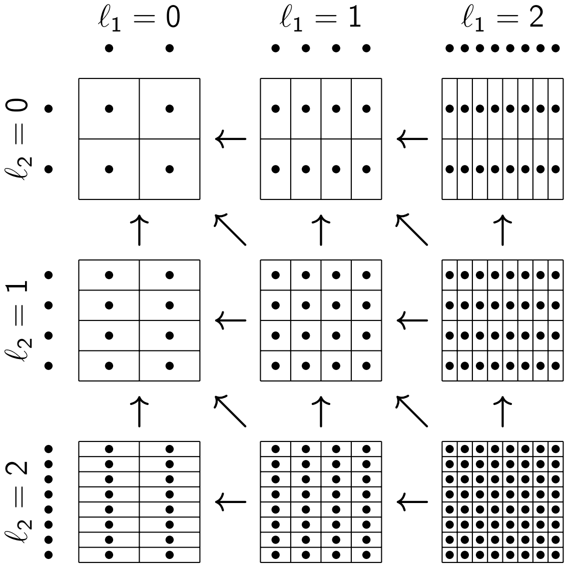
$$\Delta_i g_{\boldsymbol{\ell}} := \begin{cases} g_{\boldsymbol{\ell}} - g_{\boldsymbol{\ell}-\boldsymbol{e}_i} & \text{if } \ell_i > 0, \\ g_{\boldsymbol{\ell}} & \text{otherwise,} \end{cases} \qquad \text{for } i = 1, \ldots, m,$$

  where $\boldsymbol{e}_i$ is the i-th unit vector in $\mathbb{R}^m$
- Define multi-index difference Δ as **tensor product**

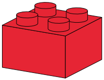
$$\Delta := \Delta_1 \otimes \cdots \otimes \Delta_m,$$

  where differences are taken with respect to all backward neighbours

A simple example

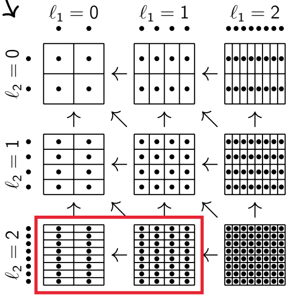


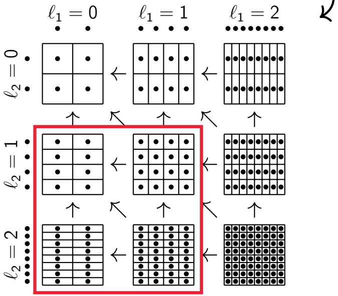
A simple example



- **Example**: suppose $m = 2$ and $\ell = (1, 2)$, then

$$\begin{aligned}
\Delta g_{(1,2)} &= \Delta_2 \left(\Delta_1 g_{(1,2)} \right) \\
&= \Delta_2 \left(g_{(1,2)} - g_{(0,2)} \right) \\
&= \Delta_2 g_{(1,2)} - \Delta_2 g_{(0,2)} \\
&= \left(g_{(1,2)} - g_{(1,1)} \right) - \left(g_{(0,2)} - g_{(0,1)} \right) \\
&= g_{(1,2)} - g_{(1,1)} - g_{(0,2)} + g_{(0,1)}
\end{aligned}$$





Multi-Index Monte Carlo

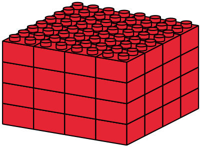
- The **MIMC estimator** for $\mathbb{E}[g]$ can be formulated as

$$Q_L(g) := \sum_{\ell \in \mathcal{I}(L)} Q\left(\Delta g_\ell\right)$$

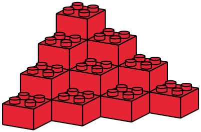
$$= \sum_{\ell \in \mathcal{I}(L)} \frac{1}{N_\ell} \sum_{n=0}^{N_\ell - 1} \left(\Delta_1 \otimes \cdots \otimes \Delta_m\right) g_\ell(\omega_{\ell,n})$$

  see [Haji-Ali, Nobile, Tempone, 2016]

- The downward closed set $\mathcal{I}(L)$ is called the **index set**
- Classical examples are



$R(\ell) := \{\vec{\tau} \in \mathbb{N}^m : \vec{\tau} \leq \ell\}$



$T_\delta(L) := \{\vec{\tau} \in \mathbb{N}^m : \boldsymbol{\delta} \cdot \vec{\tau} \leq L\}$

The optimal index set

- For a finite index set $\mathcal{I}(L)$ the error is given by

$$e(\mathcal{I}(L)) = \left| \sum_{\ell \notin \mathcal{I}(L)} \mathbb{E}[\Delta g_\ell] \right| \leq \sum_{\ell \notin \mathcal{I}(L)} |\mathbb{E}[\Delta g_\ell]|$$

- Minimize ($\sqrt{\ }$) total cost such that error is controlled

$$\begin{aligned} \min_{\mathcal{I}(L)} \quad & \sum_{\ell \in \mathcal{I}(L)} N_\ell C_\ell \\ \text{s.t.} \quad & e(\mathcal{I}(L)) \leq \text{TOL} \end{aligned}$$

- Has **no general solution** unless other assumptions on the structure of the problem are made, see [Haji-Ali, Nobile, 2016]
- Alternative strategy: build up quasi-optimal index set **adaptively** using a greedy approach

Adaptive MIMC

- Formulation as a binary (or 0-1) knapsack problem by assigning profit indicator to each index

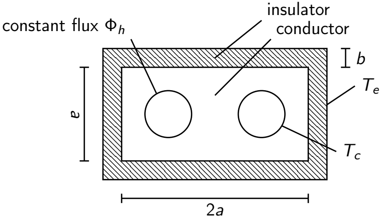
$$P_\ell = \frac{\text{error contribution}}{\text{cost contribution}}$$

$$= \frac{|\mathbb{E}[\Delta g_\ell]|}{\sqrt{\mathbb{V}[\Delta g_\ell] C_\ell}}$$

- **Objective**: find downward closed index set such that total profit is as large as possible given maximum amount of work
- Use the **active set** algorithm used in dimension-adaptive quadrature using sparse grids [Gerstner, Griebel, 2003]

Results

- Back to the example heat exchanger



- Set up an adaptive MIMC simulation with $\boldsymbol{\ell} = (\ell_1, \ell_2, \ell_3)$

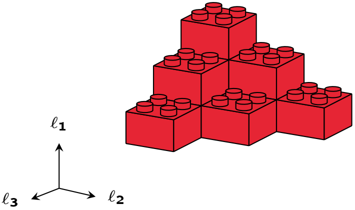
  ℓ_1 spatial discretization

  ℓ_2 number of terms in KL expansion of conductor

  ℓ_3 number of terms in KL expansion of insulator
- Number of terms in KL expansion doubles between levels
- Further algorithm details
  - index $(\cdot, 0, 0)$ corresponds to an approximation using 16 terms for conductor material and 800 terms for insulator material
  - start from index set $T_{(1,1,1)}(2)$ (simplex) to ensure robust estimates at coarser levels

Stacking bricks

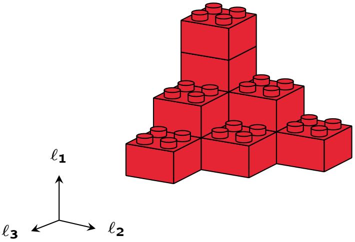
L	error[1]
10	8.4666



[1] estimated *root-mean-square error*

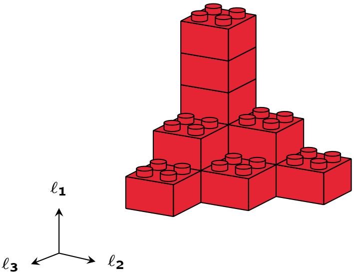
Stacking bricks

L	error
10	8.4666
11	3.4669



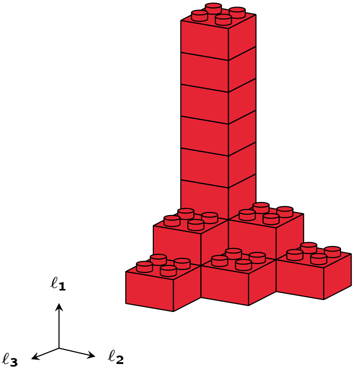
Stacking bricks

L	error
10	8.4666
11	3.4669
12	3.2260

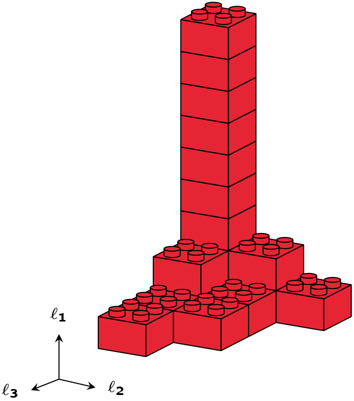


Stacking bricks

L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326

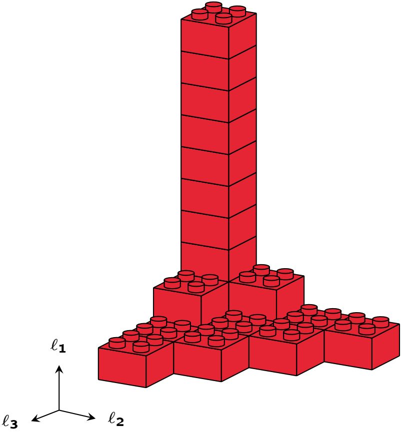


Stacking bricks



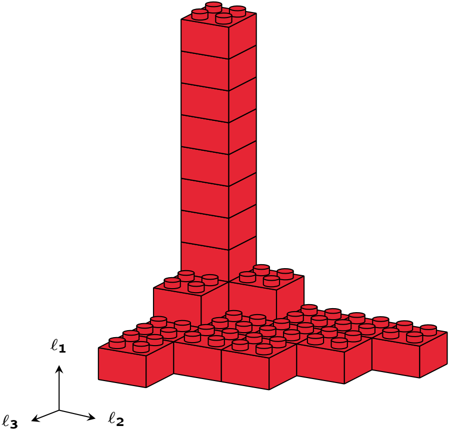
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712

Stacking bricks



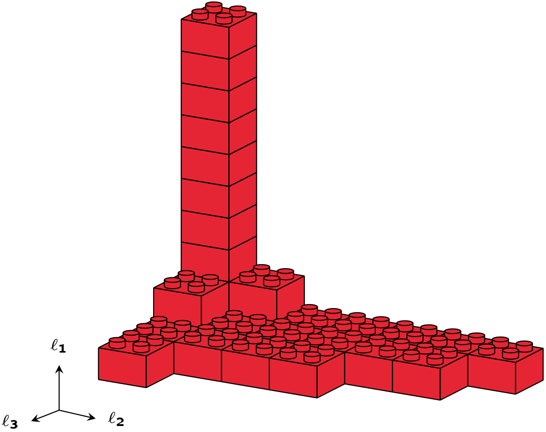
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134

Stacking bricks



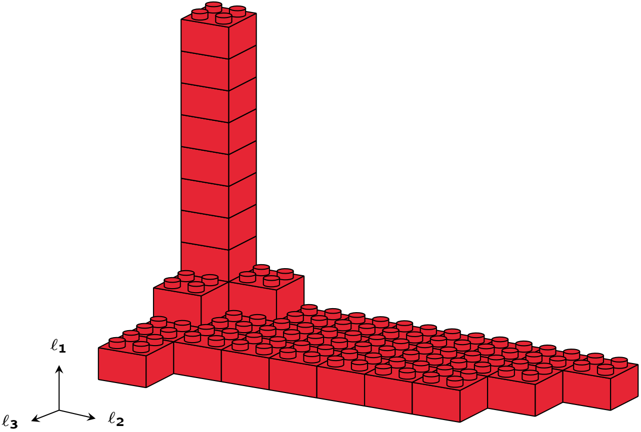
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287

Stacking bricks



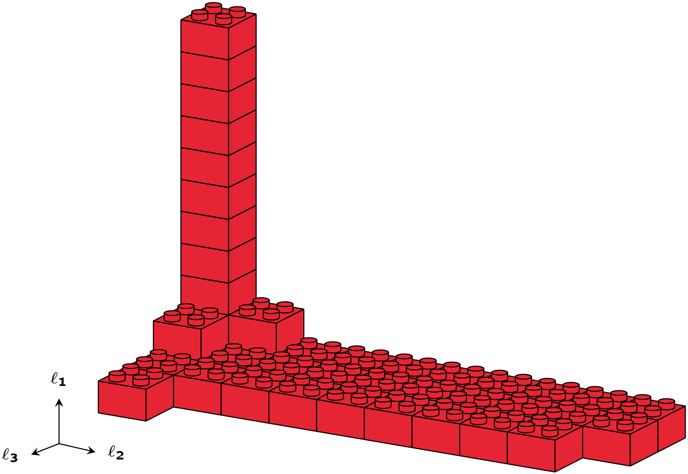
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287
28	1.5057

Stacking bricks



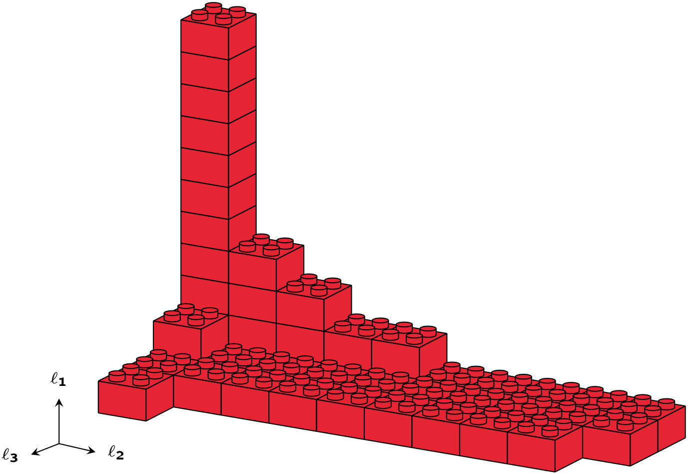
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287
28	1.5057
36	1.3762

Stacking bricks



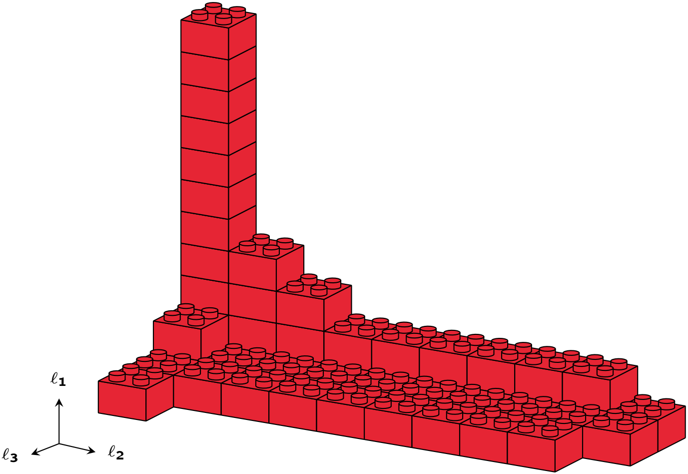
L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287
28	1.5057
36	1.3762
43	1.2319

Stacking bricks



L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287
28	1.5057
36	1.3762
43	1.2319
48	0.9681

Stacking bricks



L	error
10	8.4666
11	3.4669
12	3.2260
15	2.8326
18	2.5712
21	2.0134
24	1.6287
28	1.5057
36	1.3762
43	1.2319
48	0.9681
52	0.7543

Results

ϵ_{rel}	non-adaptive MIMC mean	RMSE	time [s]	adaptive MIMC mean	RMSE	time [s]
0.160	136.09	8.4703	39.81	136.09	8.4703	40.72 ▲
0.080	136.09	8.4666	39.92	136.09	8.4666	40.98 ▲
0.040	135.82	3.8659	50.21	136.63	3.6797	45.51 ▼
0.030	135.03	3.6890	51.93	136.08	3.4669	48.43 ▼
0.024	136.38	2.2673	78.91	136.37	3.2660	51.81 ▼
0.020	136.38	2.2673	79.90	138.07	1.6287	56.29 ▼
0.016	136.92	1.9614	132.60	138.07	1.6287	78.94 ▼
0.012	136.90	1.6252	583.90	137.74	1.5057	107.26 ▼
0.010	137.91	1.4636	4 076.46	137.93	1.3762	176.91 ▼
0.009	138.91	1.4436	4 082.30	136.68	1.2319	242.63 ▼
0.008	-	-	-	138.18	0.9681	335.27 ▼
0.006	-	-	-	138.94	0.7543	1 174.60 ▼

Extensions

What's next?

- Combination with faster sampling techniques (such as **quasi-Monte Carlo**)
  - already illustrated for non-adaptive MIMC in [R., Nuyens, Vandewalle, 2017]
  - expect significant speed-up
- Combination of MIMC with fast solvers, such as multigrid (similar to [Kumar, Oosterlee, Dwight, 2017])

Closing thoughts

- UQ for industrial applications faces **unique challenges**: dealing with high-dimensional complex models
- Use of multiple **levels** decreases computational cost of classic Monte Carlo
- We illustrated **dimension-adaptive MIMC** for approximating the expected value of a quantity of interest that is a function of the solution of a PDE with random coefficients, see [R., Nuyens, Vandewalle, 2017]
- The method does not require a priori knowledge of the structure of the problem (impossible to obtain in an industrial setting)
- Error of the adaptive index set (for fixed cost) is **smaller** compared to other classical index sets