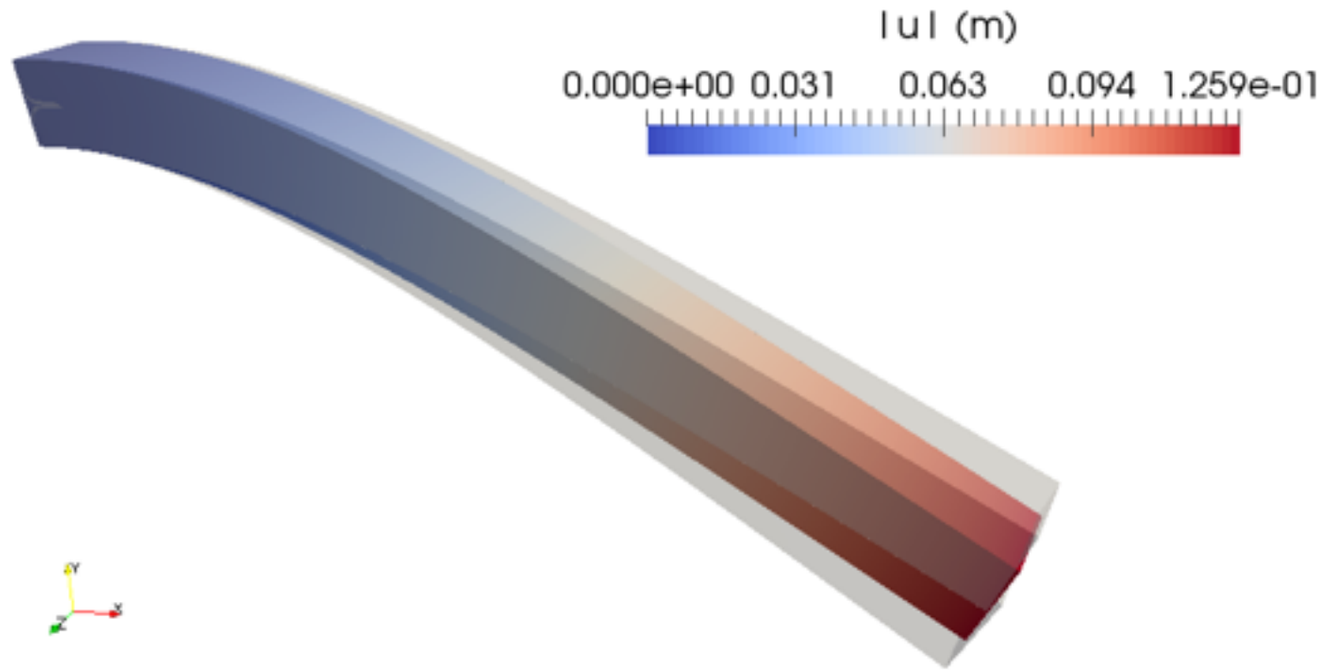
Propagating uncertainty using FE advanced Monte-Carlo methods: application to non-linear hyperelastic models

Paul Hauseux, Jack S. Hale and Stéphane P. A. Bordas

05/09/2016

Context: Soft-tissue biomechanics simulations with uncertainty

- Non-linear hyperelastic model as a stochastic PDE with random coefficients
- *Partially-intrusive* Monte-Carlo methods to propagate uncertainty



Deformation of the beam: mean +/- standard deviation

- Implementation: DOLFIN [Logg et al. 2012] and chaospy [Feinberg and Langtangen 2015]
- Ipyparallel and mpi4py to massively parallelise individual forward model runs across a cluster

1) **Monte-Carlo method**

- A non-linear stochastic system to solve can be written as:

$$F(\boldsymbol{u}, \boldsymbol{\omega}) = \mathbf{0}$$

- Expected value of a quantity of interest [Caflisch 1998]:

$$E(\psi\left(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega})\right)) = \int_{\Omega} \psi\left(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega})\right)\ dP(\boldsymbol{\omega}) = \frac{1}{Z}\sum_{z=1}^{Z} \psi\left(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega_z})\right) + o\left(\frac{||\ \psi\ ||}{\sqrt{Z}}\right)$$

Probability space: $(\Omega, \mathcal{F}, P)$

Random parameters: $\boldsymbol{\omega} = \left(\omega_1, \omega_2, \ldots, \omega_M\right)$

- The classical Monte-Carlo approach:

$$E(\psi\left(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega})\right))^{MC} \approx \frac{1}{Z}\sum_{z=1}^{Z} \psi\left(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega_z})\right)$$

2) **MC method with use of sensitivity information**

- Expected value of a quantity of interest [Cao *et al.* 2004]:

$$E(\psi(\boldsymbol{u}(\boldsymbol{x},\boldsymbol{\omega})))^{SD-MC} \approx \frac{1}{Z}\sum_{z=1}^{Z}\left(\psi(\boldsymbol{u}(\boldsymbol{x},\boldsymbol{\omega_z})) - \sum_{i=1}^{M}\frac{d\psi}{d\omega_i}(\bar{\boldsymbol{\omega}}) \times (\omega_i - \bar{\omega}_i)\right)$$

- Tangent linear model to evaluate the sensitivity derivatives [Farrell *et al.* 2013]:

$$\underbrace{\frac{\partial F(\boldsymbol{u},\boldsymbol{\omega})}{\partial \boldsymbol{u}}}_{U\times U}\underbrace{\frac{d\boldsymbol{u}}{d\boldsymbol{\omega}}}_{U\times M} = -\underbrace{\frac{\partial F(\boldsymbol{u},\boldsymbol{\omega})}{\partial \boldsymbol{\omega}}}_{U\times M}$$

U: size of the deterministic problem

M: number of random parameters

- First and Second moments of the displacement:

$$\bar{\boldsymbol{u}} \approx \frac{1}{Z}\sum_{z=1}^{Z}\left(\boldsymbol{u}(\boldsymbol{x},\boldsymbol{\omega_z}) - \sum_{i=1}^{M}\frac{d\boldsymbol{u}}{d\omega_i}(\bar{\boldsymbol{\omega}}) \times (\omega_i - \bar{\omega}_i)\right)$$

$$\bar{\boldsymbol{u}}^2 \approx \frac{1}{Z}\sum_{z=1}^{Z}\left(\boldsymbol{u^2}(\boldsymbol{x},\boldsymbol{\omega_z}) - 2\bar{\boldsymbol{u}}\sum_{i=1}^{M}\frac{d\boldsymbol{u}}{d\omega_i}(\bar{\boldsymbol{\omega}}) \times (\omega_i - \bar{\omega}_i)\right)$$

3) **Multi-level MC method with use of PCE**

- Polynomial chaos expansion (PCE) [Wiener 1936]:

$$\boldsymbol{u^k}(\boldsymbol{x},\boldsymbol{\omega}) = \sum_{\alpha \in \mathcal{J}_{m,p}} \boldsymbol{u^k}_{\alpha}(\boldsymbol{x}) H_{\alpha}(\boldsymbol{\omega})$$

- ML-MC method [Matthies 2008, Giles 2015]:

Algorithm 1 Algorithm for the multilevel Polynomial Chaos Expansion Monte-Carlo method

1: Solve the deterministic system with average parameters to obtain $\boldsymbol{u}^d$
2: $k \longleftarrow 1$
3: **while** no convergence **do**
4: **for** $z = 1$ to Z **do**
5: Generate $\boldsymbol{\omega_z} = (\omega_1^z, \omega_2^z, \ldots, \omega_M^z)$
6: Generate $\boldsymbol{u^k}(\boldsymbol{\omega_z}) = F_{pce}\left(\boldsymbol{u^{k-1}}(\boldsymbol{\omega_z})\right)$ or $\boldsymbol{u}^d$ if $k == 1$
7: Call to deterministic solver to do d (1 or more) iterations with starting values $\boldsymbol{u^k}(\boldsymbol{\omega_z})$ and all random parameter function of $\boldsymbol{\omega_z}$
8: output: $\boldsymbol{u^k}(\boldsymbol{\omega_z})$ after d iterations
9: **end for**
10: Calculate F_{pce}, the PCE of $\boldsymbol{u^k}$ from Z values of $\boldsymbol{\omega_z}$ and $\boldsymbol{u^k}(\boldsymbol{\omega_z})$
11: $k = k + 1$
12: **end while**

4) **3D Numerical simulations**

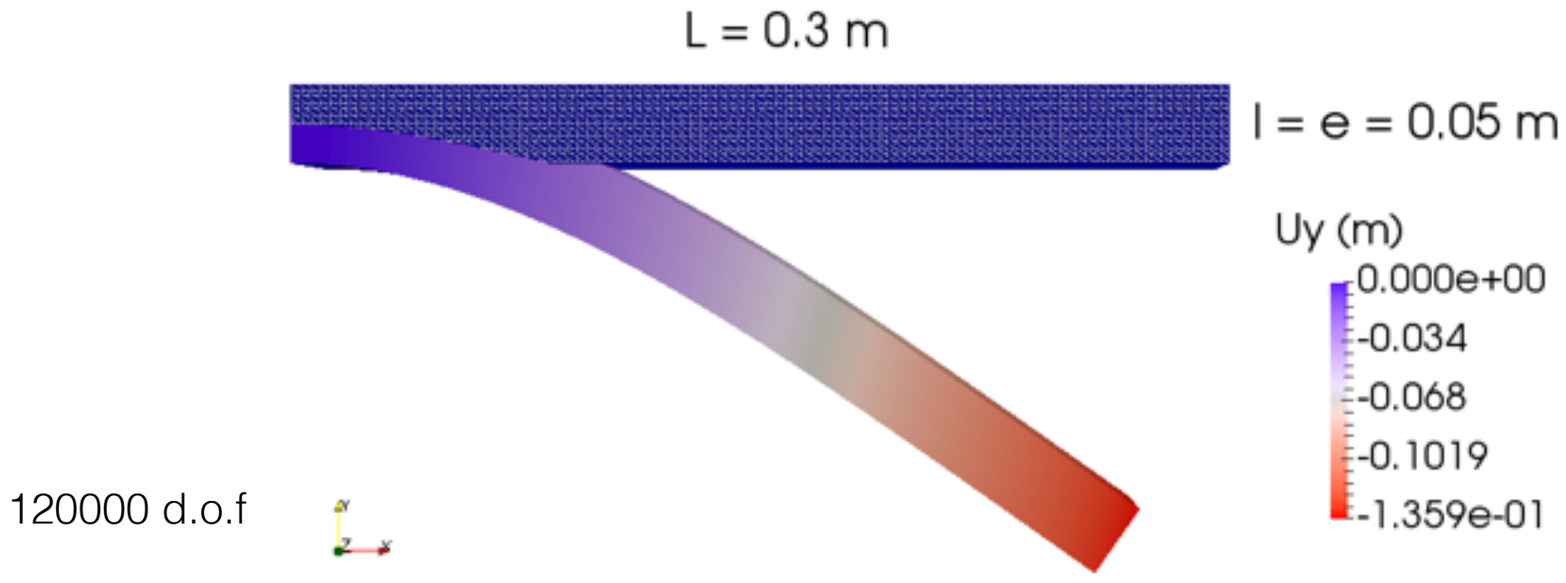


Fig: Mesh, initial configuration and deformed configuration.

- The stored strain energy density function for a compressible Mooney–Rivlin material:

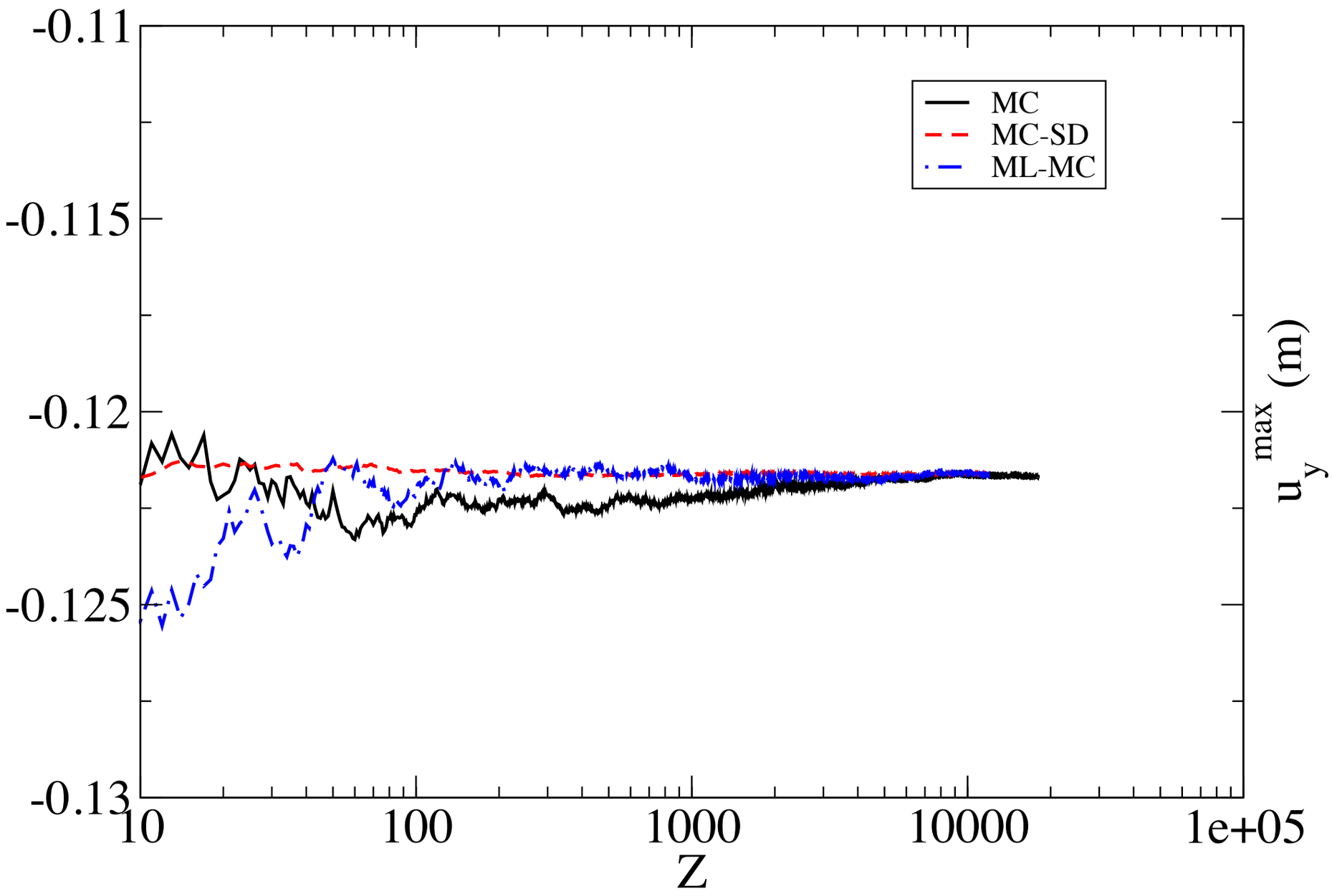
$$W = C_1(\overline{I}_1 - 3) + C_2(\overline{I}_2 - 3) + D_1(\det \mathbf{F} - 1)^2$$

- The total potential energy: $\Pi = W d\boldsymbol{x} - \rho \boldsymbol{g} d\boldsymbol{x}, \ \left(\boldsymbol{g} = g\vec{y}, g = 9.81 \ m.s^{-2}\right)$

- 2 RV with beta(2,2) distribution:

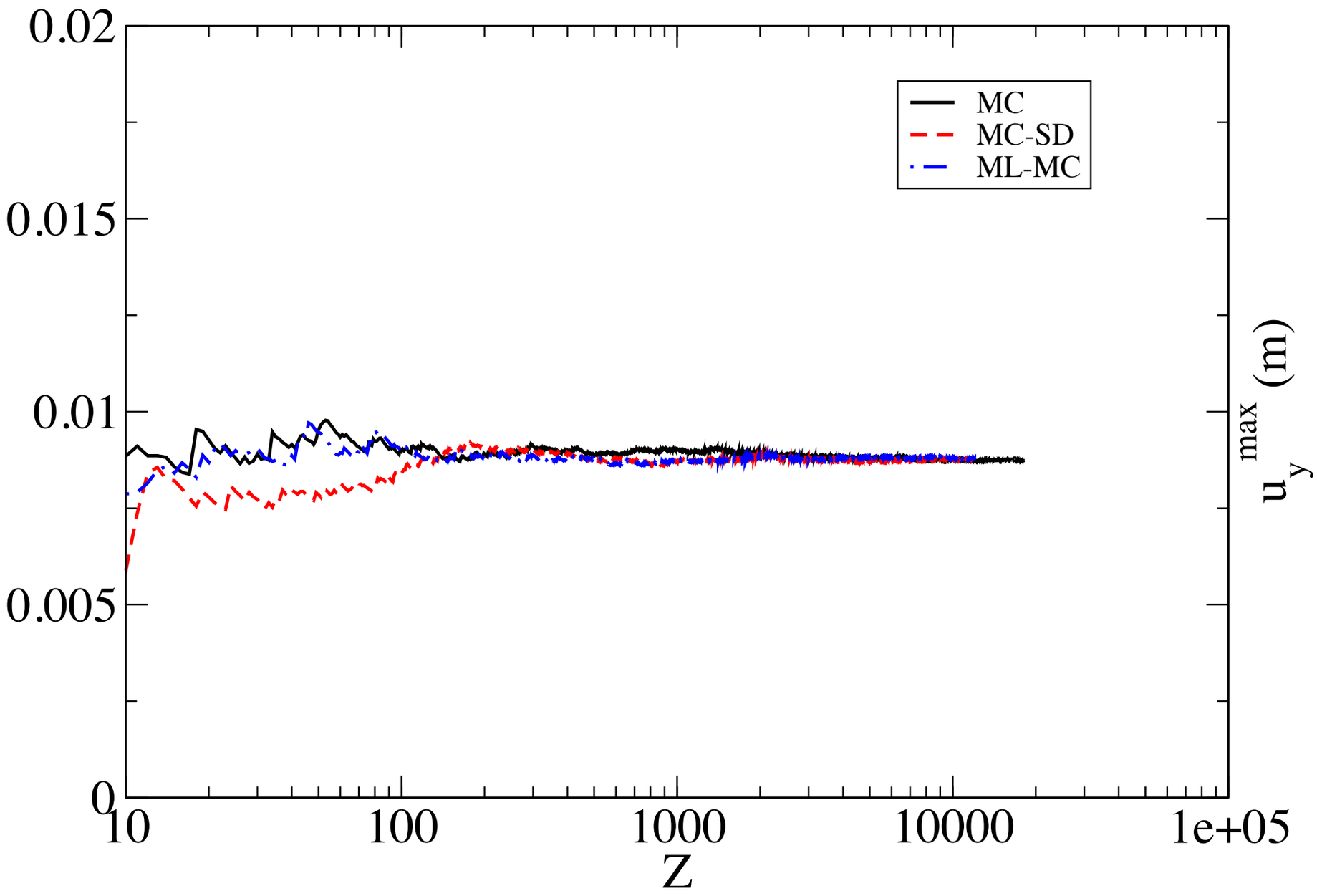
$$\begin{aligned} \rho(\omega_1) &= \rho^0(1 + \omega_1/2) \\ D_1(\omega_2) &= D_1^0(1 + \omega_2) \end{aligned} \qquad \begin{cases} D_1^0 = 2 \cdot 10^5 \text{ Pa} \\ C_2 = 2 \cdot 10^5 \text{ Pa} \\ C_1 = 10^4 \text{ Pa} \\ \rho^0 = 600 \ kg/m^3 \end{cases}$$

4) **3D Numerical simulations**



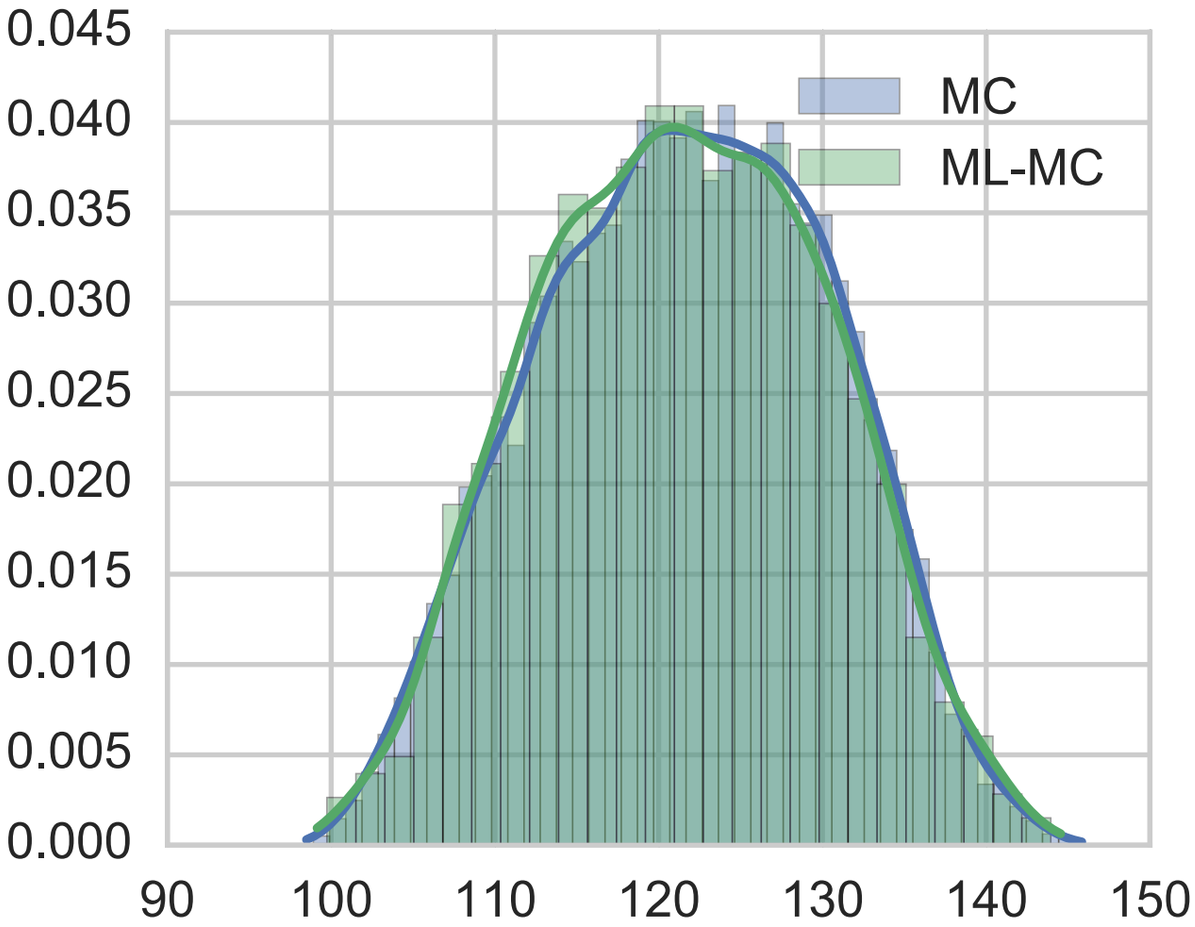
Mean

4) **3D Numerical simulations**

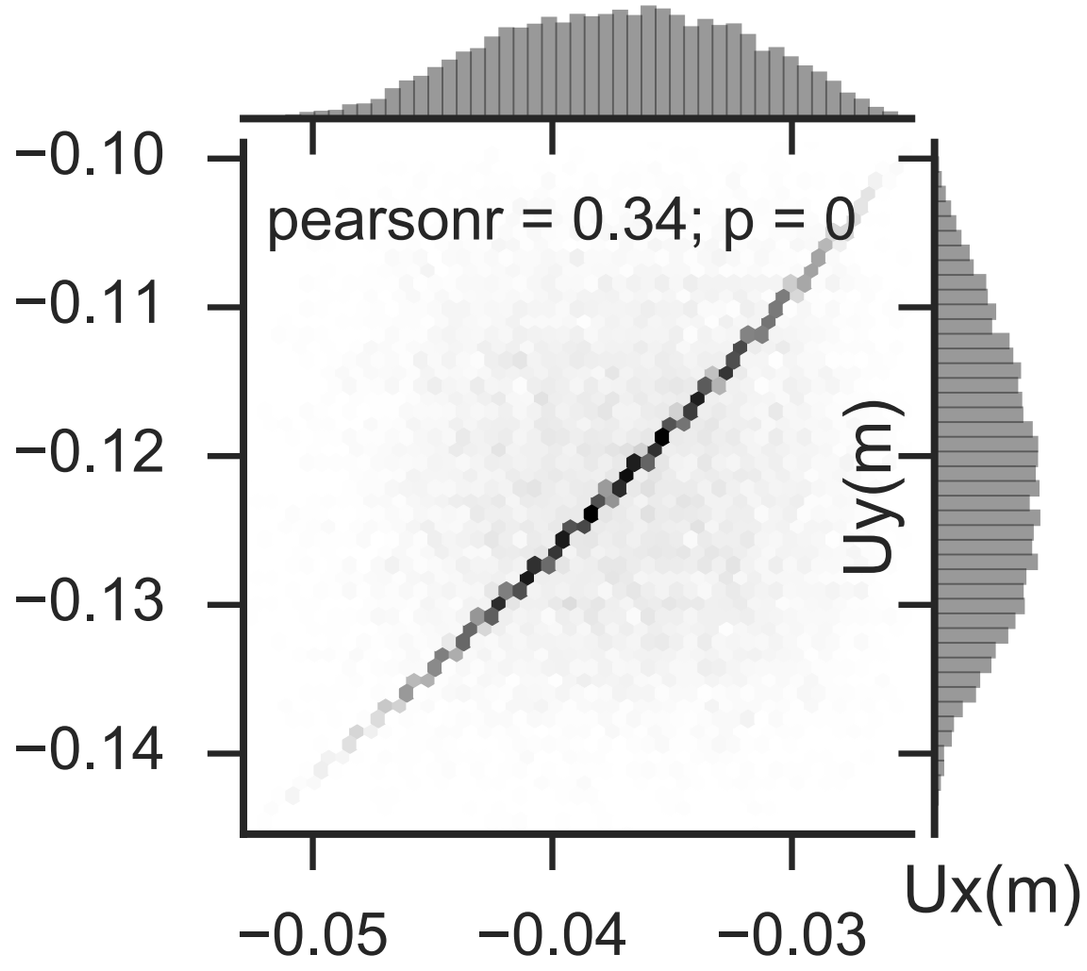


Std

4) **3D Numerical simulations**



$|u_y^{max}|(mm)$



MC-simulations

Conclusion

- *Partially-intrusive* Monte-Carlo methods to propagate uncertainty

- By using sensitivity information and multi-level methods with polynomial chaos expansion we demonstrate that computational workload can be reduced by one order of magnitude over commonly used schemes

- Implementation: DOLFIN [Logg et al. 2012] and chaospy [Feinberg and Langtangen 2015]

- Ipyparallel and mpi4py to massively parallelise individual forward model runs across a cluster