Propagating uncertainty through a non-linear hyperelastic model using advanced Monte-Carlo methods

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

05/20/2016

Stg. No. 279578 RealTCut

FEniCS workshop (FEniCS‘16) Oslo, Norway - May 18-20 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 **chaopy** [Feinberg and Langtangen 2015]
- *Ipy.parallel* 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:

\[ F(u, \omega) = 0 \]

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

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

Probability space: \((\Omega, \mathcal{F}, P)\)
Random parameters: \(\omega = (\omega_1, \omega_2, \ldots, \omega_M)\)

- The classical Monte-Carlo approach:

\[
E(\psi(u(x, \omega)))^{MC} \approx \frac{1}{Z} \sum_{z=1}^{Z} \psi(u(x, \omega_z))
\]
2) MC method with use of sensitivity information

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

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

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

\[
\underbrace{\frac{\partial F(u, \omega)}{\partial u}}_{U \times U} \underbrace{\frac{d\mathbf{u}}{d\omega}}_{U \times M} = - \underbrace{\frac{\partial F(u, \omega)}{\partial \omega}}_{U \times M}
\]

- First and Second moments of the displacement:

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

\[
\bar{u}^2 \approx \frac{1}{Z} \sum_{z=1}^{Z} \left( \mathbf{u}^2(\mathbf{x}, \omega_z) - 2\bar{u} \sum_{i=1}^{M} \frac{d\mathbf{u}}{d\omega_i}(\bar{\omega}) \times (\omega_i - \bar{\omega}_i) \right)
\]
3) Multi-level MC method with use of PCE

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

\[ u^k(x, \omega) = \sum_{\alpha \in \mathcal{J}_{M,p}} u^k_\alpha(x) H_\alpha(\omega) \]

\[ \text{dim}(\mathcal{J}_{M,p}) = (M + p)!/(M!p!) \]

- 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 \( u^d \)
2: \( k \leftarrow 1 \)
3: while no convergence do
4:   for \( z = 1 \) to \( Z \) do
5:     Generate \( \omega_z = (\omega^z_1, \omega^z_2, \ldots, \omega^z_M) \)
6:     Generate \( u^k(\omega_z) = F_{pce}(u^{k-1}(\omega_z)) \) or \( u^d \) if \( k = 1 \)
7:     Call to deterministic solver to do \( d \) (1 or more) iterations with starting values \( u^k(\omega_z) \) and all random parameter function of \( \omega_z \)
8:     output: \( u^k(\omega_z) \) after \( d \) iterations
9:   end for
10: Calculate \( F_{pce} \), the PCE of \( u^k \) from \( Z \) values of \( \omega_z \) and \( u^k(\omega_z) \)
11: \( k = k + 1 \)
12: end while
4) 3D Numerical simulations

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

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

- The total potential energy: \( \Pi = W d\mathbf{x} - \rho g d\mathbf{x} \), \( \mathbf{g} = g\mathbf{y}, g = 9.81 \text{ m.s}^{-2} \)

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

\[ \rho(\omega_1) = \rho^0(1 + \omega_1/2) \]
\[ D_1(\omega_2) = D_1^0(1 + \omega_2) \]

\[ \begin{aligned}
D_1^0 &= 2 \cdot 10^5 \text{ Pa} \\
C_2 &= 2 \cdot 10^5 \text{ Pa} \\
C_1 &= 10^4 \text{ Pa} \\
\rho^0 &= 600 \text{ kg/m}^3
\end{aligned} \]
4) 3D Numerical simulations
4) 3D Numerical simulations

![Graph showing 3D Numerical simulations with various lines representing MC, MC-SD, and ML-MC. The y-axis represents the maximum value of $u$ and the x-axis represents the Z value. The graph compares the standard deviation (Std) across different simulation techniques.](image-url)
4) 3D Numerical simulations

\[ |u_y^{max}| (mm) \]

<table>
<thead>
<tr>
<th>T (min)</th>
<th>MC</th>
<th>MC-SD</th>
<th>ML-MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>2200</td>
<td>125</td>
<td>550</td>
<td></td>
</tr>
</tbody>
</table>

Computational time with 60 engines running in parallel: comparison between the different methods with a number of realisations to have an accurate solution (MC with \( Z = 18000 \), MC-SD with \( Z = 1000 \) and ML-MC with \( Z = 6000 \)).
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


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