Flow in Random Microstructures: a Multilevel Monte Carlo Approach

Matteo Icardi1,2, Raúl Tempone2

1 Mathematics Institute, University of Warwick. E-mail: matteo.icardi@warwick.ac.uk
2 King Abdullah University of Science and Technology, Saudi Arabia. E-mail: raul.tempone@kaust.edu

Summary: In this work we are interested in the fast estimation of effective parameters of random heterogeneous materials using Multilevel Monte Carlo (MLMC). MLMC is an efficient and flexible solution for the propagation of uncertainties in complex models, where an explicit parametrisation of the input randomness is not available or too expensive. We propose a general-purpose algorithm and computational code for the solution of Partial Differential Equations (PDEs) on random heterogeneous materials. We make use of the key idea of MLMC, based on different discretisation levels, extending it in a more general context, making use of a hierarchy of physical resolution scales, solvers, models and other numerical/geometric discretisation parameters. Modifications of the classical MLMC estimators are proposed to further reduce variance in cases where analytical convergence rates and asymptotic regimes are not available. Grains can then interact arbitrarily sized objects, called grains. Grains can then interact

The computational domain is discretized with unstructured body-fitted grids, adapted locally on the grain surfaces (GMSH, snappyHexMesh). The equations are solved with Finite Element (Fenics, GetDP) or Finite Volume (OpenFOAM) methods. A detailed numerical analysis of these PDE on such complex domains is challenging but numerical tests are run to study the convergence properties of the numerical solution. Below three deterministic numerical tests (one pure diffusion problem and two Navier Stokes problems) are performed and convergence to semi-analytical reference solution is studied. Three type of grids are used: black symbols refer to uniform refinement, red to local refinement around the grains, blue to voxelized (cartesian) meshes with uniform refinement.

Multilevel Monte Carlo

Multilevel Monte Carlo techniques make cleverly use of the computational results at different refinement levels, to control and reduce the variance of the quantity of interest. Defining a quantity of interest \( Q \) (usually the integral or average of the flow variable), we can estimate its statistics \( E(Q) \) by computing with a hierarchy of levels \( \ell = 0, \ldots, L \), thus reducing the number of samples at the finer scale:

\[
E(Q) \approx \frac{1}{L} \sum_{\ell=0}^{L} E(Q_{\ell})
\]

The statistical error can be controlled by optimizing the multilevel hierarchy and the number of samples while the bias error is fully determined by the accuracy of the finest level.

Model equations

We consider a hierarchy of increasingly complex model equations

- Pure diffusion (steady)
  \[
  \Delta u(x) = 0, \quad \forall x \in \Omega \subset [0,1]^d
  \]

- Navier-Stokes (steady)
  \[
  \begin{align*}
  \nabla \cdot u &= 0, \\
  \nabla \cdot (\mu \nabla u) &= f \\
  \end{align*}
  \]

- Navier Stokes (unsteady) + volume fraction advection diffusion equation

\[
 u = \nabla \cdot \nabla \cdot \nabla u + \gamma \nabla \cdot \nabla \cdot u - \gamma \nabla \cdot (\nabla \cdot u \nabla \cdot u) = 0,
\]

Implementation and parallelization

A three-dimensional flow problem can easily take a few days of CPU time at a reasonably good resolution therefore a good parallelization strategy is needed to address complex applications (see poster II). The Monte Carlo simulation is inherently embarrassingly parallel since each sample is independent from the others. However two difficulties appear in this work. First the multilevel estimator adaptively compute the multilevel variances and mean so that a certain communication between the processors must be preserved, and the number of samples to run cannot be known in advance. For this reason the code has been first parallelized with dynamic parallel data structures (Queues) using the module rpyc to handle the communication between nodes and between processes. A second difficulty is that finer simulations need to be parallelized themselves. This second level of parallelism has been implemented using the intrinsic capabilities of the parallel solvers, with standard OpenMPI and domain decomposition techniques.

References


[4] S. Khirevich, I. Ginzburg and U. Tållarek, “Coarse-grid and non-linear solver iterations, (iv) multi-scale mathematical models and spatial averaging. The last strategy is based on the existence of homogenized or approximate macroscopic equations that are equivalent, for a given quantity of interest, to the micro-scale solution (e.g., Darcy law, eq.). The effective coefficients (e.g., permeability) of the macroscopic model however are in general unknown. Quickly computable average coefficients (e.g., permeability) of the macroscopic model how can be also introduced to represent uncertainty in the geometric parameters. These geometries define either the domain of our model equation or partition of the cubic domain characterized by heterogeneous parameters.

Discretization

Granular media and random packings

We developed a random geometry generator, developed in Python, based on a Monte Carlo random placement of arbitrarily sized objects, called grains. Grains can then interact based on algorithmic rules (e.g., Jodrey-Tory algorithm) or rigid body Newtonian dynamics and collisions (developed in the open-source code Blender) to reproduce a wide range of heterogeneous materials, from granular materials to porous media. Porosity (or, conversely, density) can be estimated a-posteriori or set as a fixed parameter. Inhomogeneities, anisotropies, fractures, and non-convex grains can be included.

Example of synthetic random packings

We developed a random geometry generator, developed in Python, based on a Monte Carlo random placement of arbitrarily sized objects, called grains. Grains can then interact based on algorithmic rules (e.g., Jodrey-Tory algorithm) or rigid body Newtonian dynamics and collisions (developed in the open-source code Blender) to reproduce a wide range of heterogeneous materials, from granular materials to porous media. Porosity (or, conversely, density) can be estimated a-posteriori or set as a fixed parameter. Inhomogeneities, anisotropies, fractures, and non-convex grains can be included.

Example of anisotropic inhomogeneous random geometry

Given a set of geometric parameters (such as domain size and grain size and shape distributions), the geometry generation is still, by construction, random but additional randomness is still, by construction, random but additional randomness

Example of anisotropic inhomogeneous random geometry (left). Convergence rate for pure diffusion problem (right).

Convergence rate for Navier Stokes with porosity 0.47 (left) and 0.366 (right).

Examples of random ellipsoid packing with geometric- and grid-based refinement.

In this work we coupled the concept of mesh-based refinement to the other refinement strategies such as (i) domain size, (ii) surface resolution and mesh quality, (iii) linear and non-linear solver iterations, (iv) multi-scale mathematical model and spatial averaging. The last strategy is based on the existence of homogenized or approximate macroscopic equations that are equivalent, for a given quantity of interest, to the micro-scale solution (e.g., Darcy law, eq.). The effective coefficients (e.g., permeability) of the macroscopic model however are in general unknown. Quickly computable average coefficients (e.g., porosity) are computed and phenomenological surrogate models (e.g., polynomial fit) are used to compute approximate effective parameters. In the case of Darcy flow, for example,

\[
 \text{Discretization} \quad \begin{align*}
 u &= \nabla \cdot \nabla \cdot \nabla u + \gamma \nabla \cdot (\nabla \cdot u \nabla \cdot u) \quad = 0, \\
 \nabla \cdot u &= 0
 \end{align*}
\]

This procedure can lead to not enough correlation between successive levels. An online correction strategy is implemented to compute the multiplicative and additive constants that minimizes the variance between two levels. This makes the statistical error estimation harder and less stable but can drastically reduce the total variance. Furthermore it can be though as a modelling tool to derive statistically correlated multiscale models.

\[
 A_{MMLC} := \sum_{\ell=0}^{L} C_{\ell} \Delta Q_{\ell}(Q_{\ell})
\]

where

\[
 \Delta Q_{\ell}(\omega) = \begin{cases} Q_{\ell}(\omega), & \text{if } \ell = 0, \\
 Q_{\ell}(\omega) - \frac{1}{C_{\ell}} Q_{\ell-1}(\omega), & \text{otherwise when } \ell > 0, 
 \end{cases}
\]

and \( C_{\ell} = 1 \). It is straightforward to show that the sum in \( A_{MMLC} \) is still stochastic, since it is equivalent to a simple substitution of each level \( Q_{\ell} \) with its scaled version \( C_{\ell} Q_{\ell} \). The ratio \( C_{\ell} \) can be chosen, as in the standard control variate technique, in such a way that the variance \( \Delta Q_{\ell}(\omega) \) is minimum.

The coarsening strategy, as in the case of the flow resistance, can also have a predictable (e.g., linear) effect on the Qol. In this case higher order extrapolation can help to reduce the bias (a la Richardson) and the variance.

\[
 A_{MMLC} := \sum_{\ell=0}^{L} C_{\ell} \Delta Q_{\ell}(Q_{\ell}) - C_{\ell-2} C_{\ell} \Delta Q_{\ell-2}(Q_{\ell-2})
\]

where now, for each sample, we have to compute a second order finite difference. \( C_{\ell} \) are now tuned to minimize the variance of the new estimator \( V_{MMLC} \) (global optimum) or recursively by minimizing first \( V_{MMLC}(Q_{\ell}) \), finding \( C_{\ell} \), then proceeding for \( \ell > 1 \).