View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by ZENODO

Available online at www.prace-ri.eu

Partnership for Advanced Computing in Europe

Large Scale Parallelized 3d Mesoscopic Simulations of the Mechanical Response to Shear in Disordered Media

Luca Marradi[a,b], Dimitris Dellis[c], and Kirsten Martens[*a,b]

[a]*Université Grenoble Alpes, LIPHY, F-38000 Grenoble, France*
[b]*CNRS, LIPHY, F-38000 Grenoble, France*
[c]*Greek Research & Technology Network, Mesogeion 56, 11527, Athens, Greece*

Abstract

In this paper we describe the development of a code that implements a coarse grained dynamics for the large scale modeleling of 3 dimensional athermal yielding and flow of disordered systems under externally applied steady shear. The stochastic lattice model for the heterogeneous flow response involves long range elastic interactions, that are resolved using fast Fourier techniques, implemented in parallel in an efficient and well scaling MPI algorithm.

1 Introduction

Flow of amorphous media is an outstanding question in two fields of science and engineering - rheology on one hand if one is dealing with soft, colloidal materials, and plasticity on the other hand for ”hard” materials, such as metallic glasses. It is a well accepted fact that these materials respond in a very heterogeneous way, i.e. fast localized particle rearrangements, plastic events, coexist with a global elastic response of the medium (see Figure 1 for a schematic view of the dynamics). On a global level we aim in this work at developing the necessary numerical tools for predicting the rheological organisation of plasticity in disordered materials undergoing various types of deformations.

Starting from the microscopic details of the yielding dynamics we follow a multi-scale approach to determine the physical phenomena that dictate the macroscopic flow behaviour. Especially when considering small-scale systems, the heterogeneous aspects of the flow will be crucial. These can only be captured in an approach that considers the microscopic reality. However simulations on the microscopic scale, such as molecular dynamics simulations are still quite limited in terms of accessible length and time scales. To overcome these limitations we implemented a lattice model on an intermediate length scale that we call mesoscopic (size of typical rearrangements in the medium, see Figure 1). This new code allows for the study of 3 dimensional geometries for which one finds very limited theoretical results in the literature. In this project we consider athermal systems, where thermal fluctuations are negligible with respect to the mechanical noise created by the dynamics itself. This is the case for example in foams, emulsions and colloids.

The results of these simulations will have a large impact on the understanding of the complex dynamics of disordered media in 3 dimensions and provide a link between the microscopic dynamics and the experimental rheological measurements in athermal systems. In this preliminary study we concentrated on the analysis of flow dynamics that show a transition towards shear banding at low enough driving, known for example in the flow of emulsions. This transition can be interpreted as a phase separation into a flowing and a non-flowing phase as shown in an earlier study in 2 dimensions [2].

*Corresponding author E-mail address : *kirsten.martens@ujf-grenoble.fr*

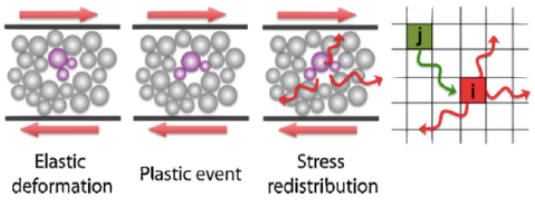


Figure 1: A scheme representing the response of jammed disordered systems to externally applied shear and its implementation into a lattice model with long range elastic interactions (taken from reference [1]).

2 Description of the code

2.1 The numerical method

We implemented a 3 dimensional lattice model (ELASTO code) for the stochastic dynamics of the stress and the flow activity in disordered systems. It is a well established point of view that flow in disordered environment occurs through local rearrangements that lead to an elastic response of the surrounding environment. These elastic interactions are long range and the stress dynamics will include convolutions of this long range elastic propagator with the plasticity field. The most efficient way to calculate these interactions is to convert the equations into Fourier space where all operations become local. We set up large scale simulations using MPI parallelization techniques for the resolution of the dynamics.

The ELASTO code is developed to investigate the nonlinear bulk response of a non Newtonian material (foams, colloid) when a finite homogeneous shear rate is applied on the system. After rescaling all relevant quantities the evolution equation for the local shear stress component appears in dimensionless form as [3]:

$$\partial_t \sigma(\mathbf{r}') = \dot{\Gamma} + \int d\mathbf{r}' G(\mathbf{r} - \mathbf{r}') n(\mathbf{r}') \sigma(\mathbf{r}) \tag{1}$$

with $\dot{\Gamma}$ the scalar shear rate (the external loading), $G(\mathbf{r}) =$ has a four-fold angular symmetry and decays as r^{-d}, where $d = 3$ is the spatial dimension as reported in Figure (2), $n(\mathbf{r})$ the activity field whose discrete values are either 1 or 0 depending if the numerical site of the grid is in a *plastic* or *elastic* state. For details of the implementation of the local yielding dynamics and the specific form of the elastic propagator see reference [2]. These dynamics introduce one additional time-scale which is the typical time $\tau_{\rm res}$ for a plastic region to restructure and to finally relax to its elatic state again. Finally, the field $\sigma(\mathbf{r})$ is the scalar shear stress component of the stress tensor.

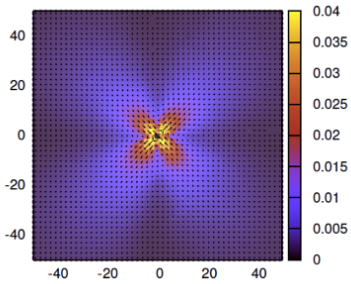


Figure 2: Averaged elastic response: The non-affine displacement field of particles induced through an imposed local plastic deformation within the bulk show a clear four fold symmetry (taken from reference [4]).

A three dimensional Cartesian box with periodic boundary conditions represents a suitable choice to reduce the numerical complexity of the code without affecting the physical phenomena like avalanche dynamics and shear band formation.

Further this choice allows for the discretisation of the interaction kernel $G(\mathbf{r})$ in Fourier space [3]. By taking the advantage in the RHS of Eq. (1) of a constant term proportional to the shear rate $\dot{\Gamma}$, and of a convolution product, a numerical pseudo-spectral method has been implemented. More precisely, the algorithms to calculate the plasticity dynamics in the elasto-plastic model include Euler steps with integrated convolutions and stochastic dynamics. The convolution, being the most time consuming part of the code, has been implemented with MPI. The choice to perform the convolution in Fourier space allow to profit from fast Fourier transform methods and convert the long range interactions, that would imply a large number of CPU communications into local, single CPU operations in Fourier space. To calculate the 3D fourier transform in parallel, a domain decomposition is applied to the Cartesian domain defined by the x, y and z directions. The parallelization occurs along two of the three directions mentioned above. An equipartition of grid points in x is made with respect to the number of processors along x. Idem for the y direction. By default the system is parallelized along the x, y directions.

2.2 The 3D FFT algorithm

Basically the idea to get a three-dimensional FFT is divided in three similar steps. For clarity the x and y directions are considered parallelized and the z one the non-parallelized. First of all, data are reorganized such that x is transposed with the z direction. Note that x becomes the non-parallelized direction. Then each processor performs multiple uni-dimensional FFTs (multi-1DFFT) along the x direction. In the second step data are once more reorganized: the y direction is transposed with the x one. The situation obtained in such a way is the same as the previous one replacing y with x. Then multi-1DFFTs along the non-parallelized directions are performed. Finally, z is transposed with the y and the algorithm is repeated. This technique, also reported in Ref. [5] PRACE white paper showed to be very efficient. The idea of the above algorithm is shown in the Figure (3).

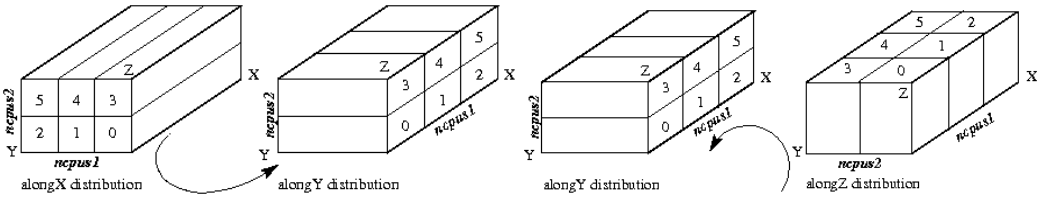
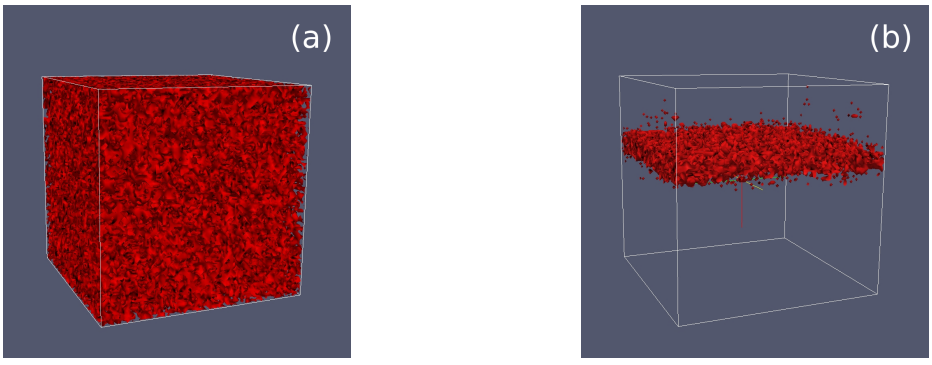


Figure 3: The figures represent the domain decomposition in the x-y plane (on the left) and in the x-z plane (on the right).



(a) (b)

Figure 4: Spatially resolved plastic activity (red color indicates plastic regions) for a long local restructuring time $\tau_{\rm res} = 10$ for a system size 64^3: **(a)** large shear rate $\dot{\gamma} = 0.1$ without shear banding and **(b)** small shear rate $\dot{\gamma} = 0.01$ with a clear signature of flow localization into a shear band.

The I/O part has been parallelized by using the the `HDF5` library [8]. To interface the output data like stress and activity with the 3D scientific data visualization Paraview, appropriate functions have been developed to print out in parallel .vti

files. The Paraview snapshot reported in Figure (4) shows the expected shear-band formation for long local restructuring times and small shear rates.

3 Scaling results

ELASTO code reported to exhibit scaling problems on PI's cluster, named Froggy. Initial runs were performed on Froggy with problem sizes 64^3, 256^3 and 512^3. In all these runs, code exhibits slow down with more than 16 cores (1 node). For the small problem size of 64^3 slow down appears from 8 to 16 cores in a single node.

The code was compiled and ran on Curie without code changes, using compilers and flags suggested in PRACE Best Practice Guides [6]. BullX MPI with Intel compilers was used. The available FFTW3 library was used to provide FFTW3 functions. The compiler optimization flags selected :

```
-O3 -xAVX -unroll -unroll-aggressive -ip
```

The main hardware difference of two machines is the network interface. Curie uses QDR [6] while Froggy uses FDR [7]. Their CPUs are similar : E5-2680v2 on Curie, E5-2670v2 on Froggy. Surprisingly, on Curie with no code changes the scaling is much better when using up to two nodes. The original code reported inverse average iteration time as function of number of cores, up to 32 cores, is presented in Figure 5a. Since code appears to exhibit better scaling on Curie, a number of runs was performed on Curie with more than 32 cores. These results are presented in Figure 5b.

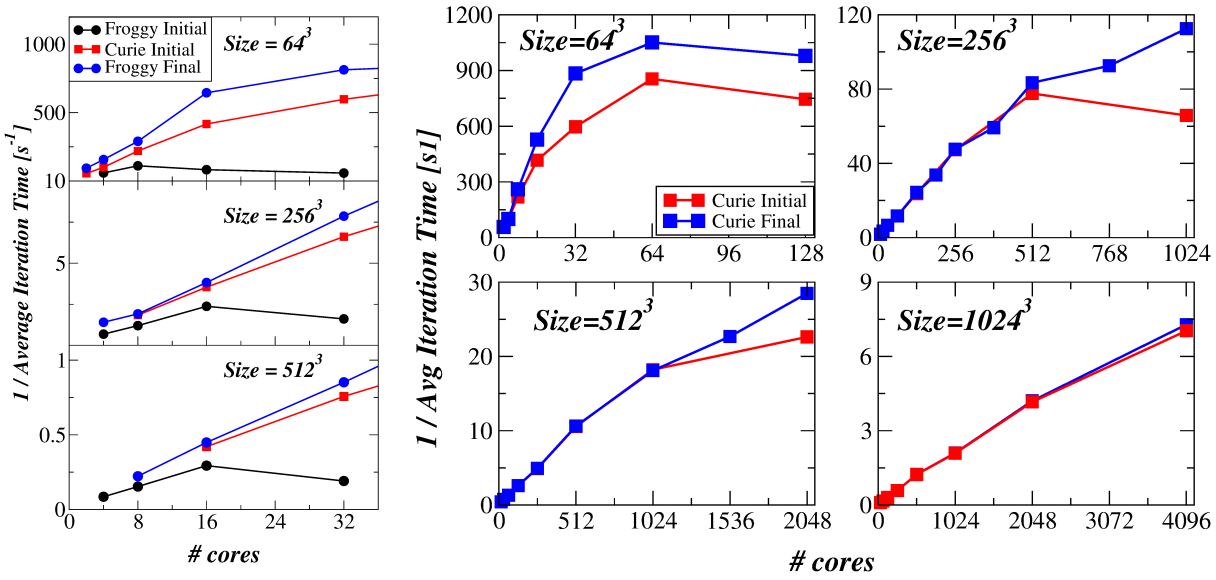


Figure 5: **(a)**. The inverse average iteration time of initial ELASTO code, using up to 32 cores, as function of number of cores on Froggy and Curie, for system sizes 64^3, 256^3 and 512^3. Depicted are the initial performance of code on Froggy (●) and Curie (■), together with the performance on Froggy after applying minor code, compiler and flags changes (●). **(b)**. The inverse average iteration time of ELASTO code as function of number of cores on Curie, for system sizes 64^3, 256^3, 512^3 and 1024^3. Depicted are the performance before (■) and after (■) applying the minor code changes.

It seems that there is a large discrepancy of the performance and scaling of the same runs on the two machines. On Curie, the code scaling looks like a typical case. On the other hand, on Froggy, going from one to two nodes leads to slow down. In addition, for single node runs, i.e. up to 16 cores, the performance on Froggy is quite lower than Curie. These findings suggest that there is some problem on PI's cluster related with network, batch system environment, MPI implementation etc. Inspecting software in use on Froggy, we found that :

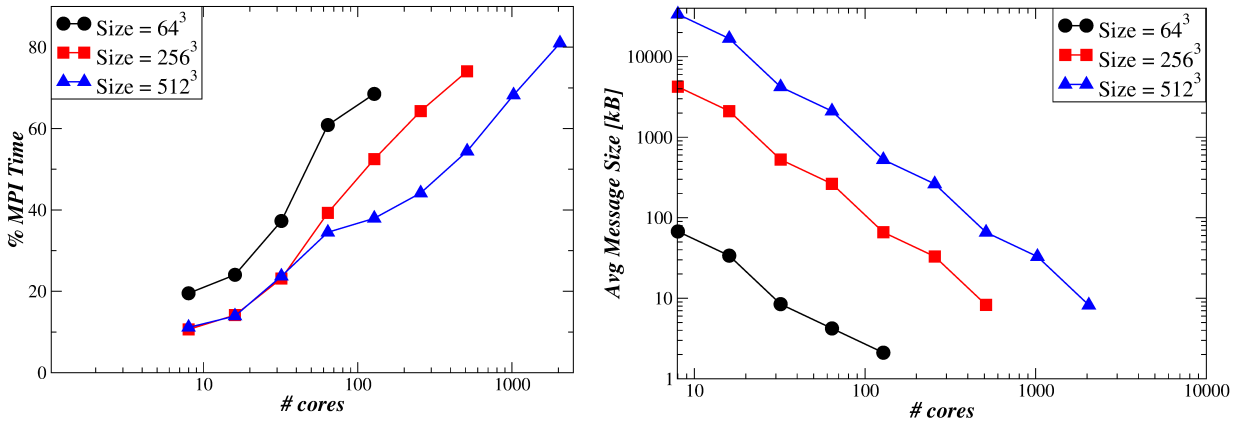
- MPI implementation is openmpi-1.6.4 with `openib` support, compiled with and using with GNU-4.6.2 compilers

- fftw-3.3 compiled with GNU-4.6.2
- Intel MPI and compiler version 13.0.1 is available on Froggy.

A number of non crucial code changes applied. Initially, all timings were performed using the fortran `cpu_time()` function. All occurencies of `cpu_time()` calls were replaced by `MPI_Wtime()` calls to get an accurate measure of the elapsed time, since `cpu_time()` measures the CPU only time of a process and is not accurate in case of load imbalancing.

3.1 Profiling

Profiling of code was performed using `Scalasca` [9] and `mpiP` [10]. In addition, some run time variables were examined inserting pieces of code at certain points. From profiling runs, the percentage of run time spent in MPI calls as well as the MPI messages size were collected. The percentage of time spent in MPI calls and the average MPI message size are presented in Figure 6.



(a) (b)

Figure 6: **(a)** The percentage of time spent in MPI calls **(b)** the Average MPI message size during run, as function of number of cores on Curie, for system size 64^3, 256^3 and 512^3.

The main conclusions from the profiling of code on Curie are summarized bellow.

- Code uses only few MPI functions. Few `MPI_Allreduce`, `MPI_BCast` and mainly point to point `send/recv` calls. When code periodically saves trajectory, for example every 1000 iterations, uses the `HDF5` library that is also uses MPI calls. These `HDF5` originated MPI calls were not profiled here.
- During a multistep run, the first iteration takes more time to complete that the rest of iterations. This fact should be taken into account, especially when one runs small number of iterations.
- During execution, on some processes and for limited number of iterations, the FFTW plane size is not identical on all processes although the deviation is not large ($\pm\ 1-3$). This introduces a small load imbalancing.
- The scalability of code depends on problem size. As shown in Figure 5b, for problem size of 1024^3 performance is almost linear up to 4096 cores. For smaller problem sizes, speedup starts to decrease after a number of cores.
- The percentage of time spent in MPI calls during run is presented in Figure 6a. The communication time increases increasing the number of cores.
- The average MPI message size as function of number of cores is presented in Figure 6b. The average MPI message size decreases by increasing the number of cores. The network performance (Bandwidth/Latency) depends on message size.

4 Conclusion and Outlook

A couple of further optimizations are planned. The first one concerns the use of OpenMP to speedup the 1DFFT. In the second one, topology structure of the cluster should be exploited in order to optimize communications between nodes as much as possible. Finally, we will exploit the fact that in the slow forcing limit, we will have to deal with rare events, that allow for the implementation of kinetic Monte Carlo steps between cascades of complex spatial-temporal dynamics. Combining standard integration methods of the equations (e.g. Euler algorithms) with kinetic Monte Carlo moves will improve the performance even further.

In conclusion we can state that we could enhance the portability of our code from the Curie cluster to the local Ciment cluster "FROGGY". Switching to intel 13.0.1 compilers and intel MPI, using the compiler flags that were used on Curie, recompiling the fftw3 library and applying the minor code changes to our local cluster FROGGY, the performance and scaling of code is close to the performance/scaling on Curie Thin Nodes. Note that for the large problem size of 1024^3 on Curie the codes scales almost linearly up to 4096 cores

After enhancing the performance of the code we are now able use the implementation of the dynamics in the future for modelling the flow of complex fluids such as foams, emulsions and colloids on a mesoscopic scale. The first tests show that the code reproduces the expected results, such as for example shear banding in the long healing time limit (see Figure 4). By developing a very general coarse grained dynamics we aim to explain the similarities in the different flow behaviours of this seemingly very different physical systems. The 3 dimensional code will be used to tackle the questions of fluctuations in the flow of disordered media, and the spatial and temporal organization of plasticity during the flow. Some of the main questions will be to search for minimal ingredients in the stochastic rules for the local yielding and relaxation processes that account for the formation of transient shear bands or permanent shear bands in the steady state regime of a steadily sheared system. In 3 dimensions it has been observed that under some conditions, within these bands, one can observe the elastic turbulence, a very complex flow behaviour that has not yet been studied within 3 dimensional elasto-plastic models. Further we will be interested in studying fluctuations in the small shear rate regime in the case of homogeneous flow. Although averaged quantities seem to indicate homogeneity, it has been shown in similar 2 dimensional systems, that in the low shear rate limit the elasto-plastic material will exhibit a complex avalanche like behaviour with intermittent bursts of succeeding plastic events. We plan to understand the statistical features of these avalanche dynamics in 3 dimensions. This understanding can be important for applications, since system spanning avalanches in amorphous solids can be interpreted as the precursors for failure and the prediction of failure for material under stress is an important field in engineering and construction.

Acknowledgements

This work was financially supported by the PRACE project funded in part by the EUs 7^{th} Framework Programme (FP7/2007-2013) under grant agreement no. RI-312763. LM acknowledges ERC grant ADG20110209 and KM acknowledges financial support of the French Agence Nationale de la Recherche, under grant ANR-14-CE32-0005 (project FAPRES).

References

[1] L. Bocquet, A. Colin, and A. Ajdari. Kinetic theory of plastic flow in soft glassy materials. *Physical Review Letters*, 103(3), 2009.

[2] K. Martens, L. Bocquet, and J.-L. Barrat. Spontaneous formation of permanent shear bands in a mesoscopic model of flowing disordered matter. *Soft Matter*, 8:4197–4205, 2012.

[3] K. Martens, L. Bocquet, and J.-L. Barrat. Connecting diffusion and dynamical heterogeneities in actively deformed amorphous systems. *Phys. Rev. Lett.*, 106:156001, 2011.

[4] F. Puosi, J. Rottler, and J. L. Barrat. Time-dependent elastic response to a local shear transformation in amorphous solids. *Physical Review E*, 89(4), 2014.

[5] M. Guarrasi, S. Frigio, A. Emerson, and G. Erbacci. Scalability improvements for dft codes due to the implementation of the 2d domain decomposition algorithm. *Prace White Paper, http://www.prace-ri.eu/IMG/pdf/wp85.pdf*, 8, 2013.

[6] Prace : Best practice guide, Curie. http://www.prace-ri.eu/img/pdf/best-practice-guide-curie.pdf, November 2013.

[7] https://ciment.ujf-grenoble.fr/wiki-pub/index.php/Hardware:FROGGY.

[8] http://www.hdfgroup.org/HDF5/

[9] Scalasca. http://www.scalasca.org.

[10] mpip: Lightweight, scalable mpi profiling. http://mpip.sourceforge.net/.