Accelerating Yade's poromechanical coupling with matrix factorization reuse, parallel task management, and GPU computing

Robert A. Caulk, Emanuele Catalano, Bruno Chareyre

Version of Record: https://www.sciencedirect.com/science/article/pii/S0010465519303340
Manuscript_8355b8e0b7d952653fe9f625b5e0061d

Accelerating Yade's poromechanical coupling with matrix factorization reuse, parallel task management, and GPU computing

Robert Caulk, Emanuele Catalano, Bruno Chareyre

Univ. Grenoble-Alpes, CNRS, Grenoble INP, 3SR Lab, F-38000 Grenoble, France

1. Abstract

This study details the acceleration techniques and associated performance gains in the time integration of coupled poromechanical problems using the Discrete Element Method (DEM) and a Pore scale Finite Volume (PFV) scheme in Yade open DEM software. Specifically, the model is tailored for accuracy by reducing the frequency of costly matrix factorizations (matrix factor reuse), moving the matrix factorizations to background POSIX threads (multithreaded factorization), factorizing the matrix on a GPU (accelerated factorization), and running PFV pressure and force calculations in parallel to the DEM interaction loop using OpenMP threads (parallel task management). Findings show that these four acceleration techniques combine to accelerate the numerical poroelastic oedometer solution by 170x, which enables more frequent triangulation of large scale time-dependent DEM+PFV simulations (356 thousand+ particles, 2.1 million DOFs).

To be submitted to the Journal of Computer Physics Communications

2. Introduction

The poroelastic behavior of geomaterials has become a focus of modern geomechanical research (Detournay and Cheng, 1993; Wang, 2017) due to its significance in real world scenarios like dam failures or deep geo-energy reservoirs (Zoback, 2007). One branch of poroelastic research focuses on elucidating grain and pore scale processes through numerical modeling (e.g. Edwards et al. (1991); Willingham et al. (2008)). But despite careful tailoring

of these poroelastic numerical models for stiff particulate systems and non-turbulent flow (Chareyre et al., 2012), the time-dependent implicit flow problem becomes unmanageably time consuming beyond 70 thousand particles on an office workstation (ca. 500 thousand degrees of freedom). The present study attempts to remedy some implicit solver weaknesses by demonstrating the implementation and performance of matrix factor reuse, multithreaded factorization, GPU accelerated factorization, and parallel task management in the Pore Finite Volume (PFV) scheme coupled to a Discrete Element Method (DEM) model.

The DEM+PFV model is particularly well suited for the simulation of poroelasticity in geomaterials since DEM is well established for particulate modeling (Cundall and Strack, 1979; O'Sullivan, 2011), including soil (Zhu et al., 2008), rock (Scholtès and Donzé, 2012), and concrete (Camborde et al., 2000). Further, DEM couples naturally with PFV since DEM particles double as tetrahedral nodes in a PFV triangulation. These characteristics create a highly efficient poroelastic coupling, capable of helping researchers simulate accurate poroelastic processes at a fraction of the computational price of traditional Finite Element + CFD couplings (Chareyre et al., 2012). One DEM software in particular, Yade open DEM, was poroelastically coupled by triangulating DEM particle locations to generate a pore network, which enabled both the estimation of fluid forces on particles as well as pore volume changes due to particle movements. After Yade's unique poroelastic model was validated using an oedometer test (Catalano et al., 2014), it was extended for other DEM poroelastic applications such as hydraulic fracturing (Papachristos et al., 2017) and multi-phase flow (Chalak et al., 2017; Yuan et al., 2017; Sweijen et al., 2018). Although these applications are shedding light on grain scale poroelasticity, Yade's pore finite volume (PFV) scheme is still constrained by frequent refactorizations of large matrices associated with its time-dependent implicit flow solution. For example, the practical simulation of hydraulic fracture requires matrix refactorization for each explicit time-step that involves fracture propagation. For these types of simulations, DEM's explicit time-stepping scheme is no longer the bottleneck, instead it is the factorization of the implicit PFV scheme conductivity matrix. Further, the

computational expense of the PFV coupling is exacerbated in viscous dominated systems where the maximum allowable timestep decreases compared to a dry system (Appendix A). For these reasons, various acceleration techniques are presented within this paper to alleviate aforementioned weaknesses.

Many acceleration and parallelization techniques already exist for both DEM and the solution of linear systems of equations. DEM benefits from an easily parallelizable explicit time integration that is typically accelerated using OpenMP and MPI methods (Weatherley et al., 2011; Šmilauer and Chareyre, 2015). Meanwhile, the solution of linear systems of equations, like the PFV scheme accelerated herein, is usually accelerated depending closely on the sparsity and symmetry of the "stiffness" matrix (referred to as the "conductivity matrix" herein). In most linear FEM cases, for instance, the system is sparse and symmetric, but the stiffness matrix requires inversion at each time step (Smith et al., 2013). Typically parallelizable iterative solvers, such as conjugate gradient, can be employed in OpenMP (Ju, 2010), MPI (Jimack and Touheed, 2000), or GPU (Liu et al., 2008; Kakay et al., 2010). These stiffness matrices are generally preconditioned to accelerate the solution (Chen, 2005). Direct solvers employ matrix factorization methods as an alternative solution, which enables the reuse of a single factorization for multiple right hand sides if the system is defined by the same conductivity matrix over multiple time iterations (Booth et al., 2011). If the rank change of the conductivity matrix is low, acceleration to solution can be found by updating/downdating the factor (Davis and Hager, 2009). The PFV solver presented here relies on a direct solution which opens up several acceleration techniques. Thus, the objective of this study is to increase the performance of the coupled DEM+PFV model by introducing four acceleration techniques, including: 1) reusing matrix factorizations for multiple flow time steps, 2) moving matrix factorization to a separate background thread, 3) reducing the cost of matrix factorization by applying GPU acceleration, and 4) computing PFV pore pressures, volumes, and fluid forces in parallel with the DEM interaction loop.

3. Methods

3.1. Discrete Element Method (DEM)

DEM treats particulate material as an assembly of various sized spheres, each characterized by density and stiffness (Cundall and Strack, 1979). Spherical particle interactions and movements are governed by Newton's second law of motion:

$$m_i \ddot{\mathbf{x}}_i = \mathbf{f}_i \tag{1}$$

where $\ddot{\mathbf{x}}_i$ is the acceleration of particle i, m_i is the particle mass, and $\mathbf{f_i}$ is the particle traction. Traction is estimated as shown below using the trajectories of all particles interacting with particle i. Particle trajectories and positions are estimated using an explicit time stepping scheme which uses the particle acceleration from the current step in addition to the velocity from the previous step (Šmilauer and Chareyre, 2015). Once the next-step position of all particles are approximated, the new particle overlap (ΔD) is used as a strain evaluation in the estimation of inter particle normal and shear ($\mathbf{f}_s$) forces:

$$\mathbf{f}_{n,ij} = k_{n,ij} \Delta D_{ij} \cdot \mathbf{n}_{n,ij} \tag{2}$$

where ($\mathbf{f}_{n,ij}$) is the normal force between particles i and j, $k_{n/s}$ are the normal and shear stiffnesses, ΔD_{ij} is the displacement between particles, and $\mathbf{n}_{n,ij}$ is the unit vector parallel to the interaction between particles. Since the shear force depends on the orientation of both particles, it is updated incrementally:

$$\Delta \mathbf{f}_{s,ij} = k_{s,ij} \Delta u_{s,ij} \cdot \mathbf{n}_{s,ij} \tag{3}$$

$$\mathbf{f}^t_{s,ij} = \mathbf{f}^{t-\Delta t}_{s,ij} + \Delta \mathbf{f}_{s,ij} \tag{4}$$

where $\mathbf{f}^t_{s,ij}$ is the shear force between particles i and j at time step t, $\mathbf{n}_{s,ij}$ is the unit vector perpendicular to the particle interaction, $\Delta u_{s,ij}$ is the tangential displacement and $k_{s,ij}$ is

simply a fraction of $k_{n,ij}, (k_s/k_n)$. Finally, the traction on a particle i interacting with n neighbors becomes:

$$\mathbf{f}_i = \sum_{j=1}^{n} \left(\mathbf{f}_{n,ij} + \mathbf{f}_{s,ij} \right) \tag{5}$$

which is used in the time integration of Eq. 1.

3.2. Pore Finite Volume (PFV) Scheme

Yade's PFV scheme was introduced by Catalano et al. (2011), Chareyre et al. (2012), and Catalano et al. (2014). Refer to Chareyre et al. (2012) for a thorough description of the poroelastic model, the pore network, and fluid-particle force approximations. In summary, the Discrete Element sphere locations are regular delaunay triangulated to form a tetrahedral mesh. Each tetrahedral is comprised of four discrete elements and represents a single pore comprised of solid and fluid fractions. The total network of tetrahedrals constitutes a pore network, which is used to establish a Stokes-flow. Assuming small Reynolds and large Stokes numbers, the continuity equation can be written as a surface integral:

$$\dot{V}_{p,i} = \int_{\partial\Theta_i} (\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} \; dS \tag{6}$$

where $\dot{V}_{p,i}$ is the pore volume change, $\partial\Theta_i$ is the pore contour, and $\mathbf{u}$ is the fluid velocity relative to the contour velocity $\mathbf{v}$. Since the solid area of the pore will not change, $\partial\Theta_i$ can be reduced to only the fluid fractions (S_{ij}^f) of the pore contour. Thus, the integral can be represented as the sum of fluid fluxes exchanged by each pore and its four neighbors (j=1 to 4):

$$\dot{V}_{p,i} = \sum_{j=1}^{4} \int_{S_{ij}^f} (\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} \; dS = \sum_{j=1}^{4} q_{ij}. \tag{7}$$

Flux (q_{ij}) through the pore throat connecting pore i and j is approximated by the local pressure gradient:

$$q_{ij} = k_{ij}\frac{p_i - p_j}{l_{ij}} \tag{8}$$

where p_i and p_j are the pressures of neighboring pores and l_{ij} is the length of the connecting pore throat. The hydraulic conductance, $g_{ij} = k_{ij}/l_{ij}$, can be approximated using Poiseuille, the details of which can be found in Chareyre et al. (2012).

Finally, a linear system can be constructed based on the pressure at time $t + \Delta t$ as a function of the volume changes at t:

$$\sum_{j=1}^{4} g_{ij}\left(p_i^{[t+\Delta t]} - p_j^{[t+\Delta t]}\right) = \dot{V}_{p,i}^{[t]} + Q_i^{[t]} \tag{9}$$

where Q_i is a source term for pore i. The matrix representation of the full linear system is simply the known conductivity matrix $\mathbf{G}$ comprising the g_{ij} coefficients from Eq. 9 for all i, the unknown pressures listed in a vector $\mathbf{p}$, and the vector of rate of volume changes $\dot{\mathbf{V}}$. $\dot{\mathbf{V}}$ depends linearly on particles velocities, which can be expressed by an operator $\mathbf{E}$ such that $\dot{\mathbf{V}} = \mathbf{E}\dot{\mathbf{x}}$. The instantaneous pressures-velocities relation finally reads:

$$\mathbf{G}\mathbf{p} = \mathbf{E}\dot{\mathbf{x}} + \mathbf{Q} \tag{10}$$

$\mathbf{G}$ is sparse, symmetric, and positive definite (shown in Figure 1). Therefore, Cholesky decomposition is employed for the decomposition of $\mathbf{G}$ to a lower triangular matrix multiplied by its transpose ($\mathbf{L}\mathbf{L}^T$). The decomposed matrix, i.e. the factor, can be used to solve for $\mathbf{p}$ by first using forward substitution followed by back substitution:

$$\mathbf{L}y = \dot{\mathbf{x}} \tag{11}$$

$$\mathbf{L}^T\mathbf{p} = y \tag{12}$$

thus avoiding the prohibitively expensive inversion of $\mathbf{G}$ for the solution of $\mathbf{p}$. The drag forces on the particles ($\mathbf{f}_D$) are obtained after multiplication of the pressure vector by a matrix $\mathbf{F}$

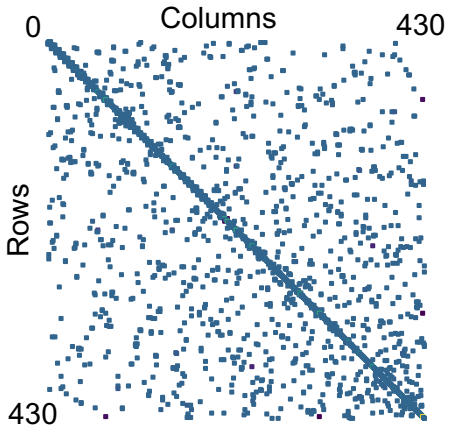

Figure 1: Example of a 430 DOFs positive definite, symmetric, banded, sparse conductivity matrix ($\mathbf{G}$)

whose components reflect projected area:

$$\mathbf{f}_D = \mathbf{F}\mathbf{p} \tag{13}$$

As discussed and quantified throughout the remainder of the paper, the computational expense of the poroelastic DEM+PFV coupling is not insignificant. However, the introduction of poroelasticity can compound the computational slowdown by also reducing the maximum stable time step. As demonstrated in Appendix A, as soon as typical DEM stiffness effects (the natural period of a spring mass system) become negligible compared to viscous effects (fluid drag forces acting like dampers) the maximum time step depends on the maximum eigenvalue of the viscous system. It is not uncommon for a poroelastic simulation of granular material to operate at a time step equal to one order of magnitude lower than its dry counterpart. Thus, the need for the acceleration techniques highlighted herein is even more pertinent.

4. Acceleration techniques

4.1. Matrix Factor Reuse

Catalano (2012) showed how the factorization of [$\mathbf{G}$] consumes ca. 98% of the total flow

solver time. In comparison, the simple process of forward and back substitution into the factor for the solution of $\{\mathbf{P}\}$ is negligible. For this reason, total factorizing is reduced by reusing the costly factor for multiple right-hand solves (refer to Figure 3 to see the relationship of matrix factor reuse to the rest of Yade's DEM+PFV algorithms). In other words, as long as the deformation criterion (Eq.14) is satisfied, the factor is reused for the duration of a remesh interval, λ_{rm}. This factor-reuse reduces the cost of determining $\{\mathbf{P}\}$ by an order of magnitude since the expensive factorization is not repeated. The negligible effect of remesh interval during a quasi-static geomechanical oedometer test is confirmed by comparing pressure at the same location and time (Sec.5.2) for nine different remesh intervals (Fig. 2a). Pressure differences are negligible and random, owing to the effect of force summation order in parallel environments for DEM, as shown by the replicate rests run for Fig. 2b. Both analyses demonstrate how matrix factor reuse does not significantly impact the solution of the quasi-static oedometer simulation used for performance benchmarking throughout the remainder of this paper.

In dynamic simulations associated with large deformations, the remesh interval depends on deformation criteria. For instance, the criterion

$$\max(\varepsilon_{v,i}^{t_0 \to t}) < 0.01 \tag{14}$$

can be used where $\varepsilon_{v,i}^{t_0 \to t}$ is the volume change of pore i since last remesh. Remeshing would be triggered when that condition is not satisfied. Auxiliary analyses compared this remesh criterion to remeshing at each interval and concluded that geometrical and mechanical variables are sufficiently representative of the state of the medium during deformation, to yield accurate results.

4.2. Multithreaded factorization

Despite accelerating the solution, the matrix factor reuse scheme described in Sec. 4 still requires the DEM simulation to stop at the end of a remesh interval and perform both the retriangulation of the pore network and the factorization of $[\mathbf{G}]$. During this interim step, the

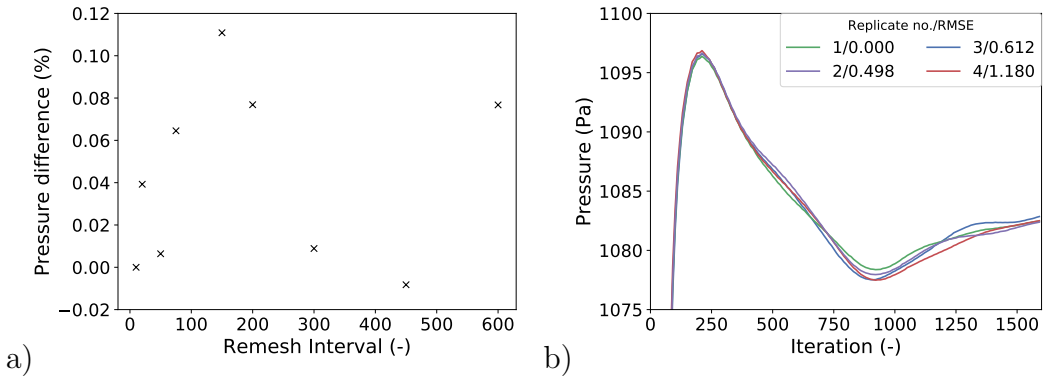

Figure 2: a) Pressure difference for various remesh intervals after 1200 iterations and 10 parallel cores for DEM force summations b) Replicate tests using Remesh interval = 20 iterations and 10 parallel cores for DEM force summations

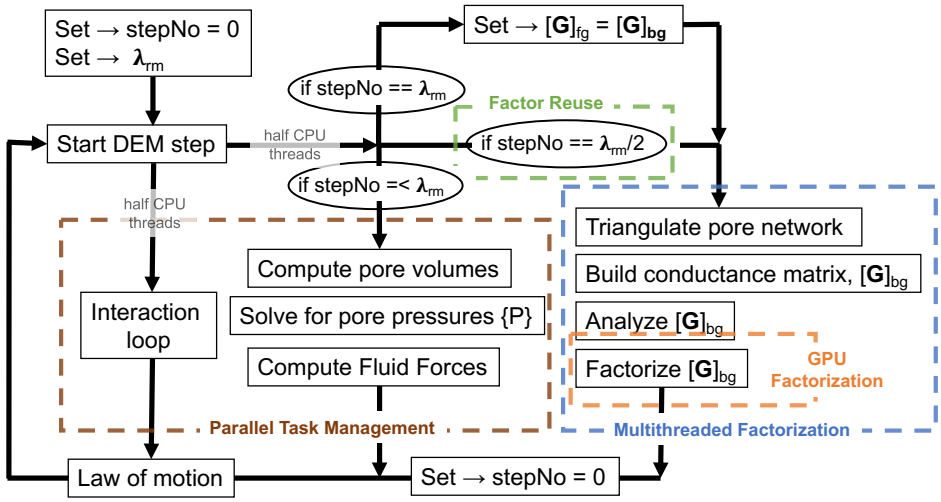

Figure 3: Yade DEM+PFV accelerated algorithm overview.

DEM+PFV simulation cannot continue stepping through time since it needs to wait for the new conductivity matrix before it can obtain pore pressures and the associated viscous and pressure forces. To address this weakness, a multithreaded scheme was added to Yade's PFV with the objective of retriangulating the pore network and factorizing [**G**] on background POSIX threads while the DEM+PFV simulation steps forward with a previous pore network and prefactorized [**G**] on foreground OpenMP threads (refer to Algorithm 1 and Figure 3). This multithreaded configuration will improve performance for all simulations associated with any λ_{rm}, but there exists an optimal λ_{rm} that will yield *uninterrupted* time stepping through the coupled DEM+PFV simulation provided the time required to retriangulate the pore network and factorize [**G**] is less than the time it takes the coupled DEM simulation to step through $\lambda_{rm}/2$ steps. In other words, the optimal λ_{rm} for uninterrupted simulation is dictated by the speed of the simulation (v_{iter}, iter/sec) and the background time (t_{bg}, s):

$$\lambda_{rm} \geq 2 t_{bg} v_{iter} \tag{15}$$

4.3. GPU Accelerated Factorization

The present study aims to reduce the heavy cost of [**G**] factorization in Eq. 10 by leveraging GPU computing. In particular, the PFV scheme presented here employs 'CHOLMOD', a GPU accelerated sparse matrix solver part of the open source SuiteSparse C library (Davis, 2013). CHOLMOD provides Cholesky decomposition, it builds an elimination tree of the matrix based on a METIS partitioning, and sends subtrees directly to the GPU for factorization (Rennich et al., 2016). The subtree algorithm is highly optimized to reduce the volume of data exchange between the GPU and the CPU.

4.4. Parallel Task Management

The final acceleration technique, called Parallel Task Management (Figure 3), exploits the highly parallel nature of DEM's interaction detection and force collection methodologies. Since the time integration of particle movement depends solely on the traction from the

Algorithm 1 Multithreaded triangulation and factorization

$simulationRunning \leftarrow$ simulation activity boolean
$stepNo \leftarrow$ number of steps since last remesh
$\lambda_{rm} \leftarrow$ remesh interval
Foreground simulation $(_{fg})$
while $simulationRunning = True$ **do**
  foreground OpenMP threads solve for pore pressure at each time step by reusing:
  $triang_{fg} \leftarrow$ foreground pore network
  $factor_{fg} \leftarrow$ foreground factorization
  Background factorization $(_{bg})$
  if $stepNo = \lambda_{rm}/2$ **then**
    background POSIX threads retriangulate pores and build/factor conductivity matrix:
    $triang_{bg} \leftarrow$ retriangulate pore network
    $factor_{bg} \leftarrow$ factorize conductivity matrix
  end if
  if $stepNo = \lambda_{rm}$ **then** set new bg solver to fg:
    $triang_{fg} = triang_{bg}$
    $factor_{fg} = factor_{bg}$
    $stepNo = 1$
  end if
end while

current time step (Eq. 1), fluid forces can be collected in parallel just like the particle-particle forces are collected in parallel. As shown in Figure 3, the fluid force algorithm is initiated on a separate set of OpenMP threads from the contact detection threads. DEM forces and fluid forces are combined before the final integration step.

5. Test setup

5.1. Computer Details

All simulations presented in this study were performed on a scientific workstation containing the following hardware:

- **CPU** Xeon 2680 v2 E5 2.8 GHz 10 core processor, 448 GFLOPS double precision
- **GPU1** GeForce 1050 Ti, 4 GB RAM, 1392 MHz, 32 cuda cores, 61.9 GLOPS double precision

- **GPU2** Tesla K20, 5 GB RAM, 2496 MHz, 706 cuda cores, 1175 GFLOPS double precision, ECC=ON
- **RAM** 64 GB 1866 MHz
- **Storage** 500 gb SSD 600 MB/s read/write

and the following software:

- Linux Ubuntu 18.04
- Yade git-28917a9
- OpenMP parallelization
- SuiteSparse 4.6.0-beta
- CUDA 9.0
- Nvidia 384.11 GPU drivers

5.2. Model details

The DEM+PFV performances of multi-core CPU, GeForce 1050 Ti GPU, and Tesla K20 GPU conductivity matrix factorizations (Eq.12) were evaluated using a pre-validated (Catalano et al., 2014) consolidation test of a saturated soil packing (example script [1]). The DEM sphere packing is cubically sized from 8e-6 to 3.4e-3 m^3 (Figure 4) with microparameters as shown in Table 1. The fluid and mechanical boundary conditions follow traditional oedometer boundary conditions as shown in Figure 4: enclosing walls impose a deviatoric stress of 1 kPa (Neumann) in the Y direction and maintain fixed displacement (Dirichlet) in the X and Z directions. Meanwhile, fluid boundary conditions include drained (Dirichlet - imposed pressure of 0 Pa) at the top Y cube face and impermeable (Neumann - no flux) on the remaining cube faces. All flow is calculated using the dynamic viscosity of water μ=1 cP. Both mechanical and fluid time steps are set constant to 1e-6 s and the simulation proceeds for 600 time steps with λ_{rm}=200.

[1] GitHub: yade/trunk/examples/oedometer.py

5.3. Data description

A parametric sweep was performed for three device types and six problem sizes, resulting in 18 total simulations. For each parametric combination, six distinct timings were collected and averaged for each of the following seven algorithms:

(1) Build the system of linear equations

(2) Allocate the system to memory

(3) Analyze the system (identifiy non-zero pattern and build elimination tree)

(4) Factorize the system ([**G**] matrix decomposition)

(5) Solve the system (forward/backward substitution into factor)

(6) Compute pore volumes

(7) Compute fluid forces (pressure and viscous forces)

Additionally, the simulation speed and total time to step 600 iterations of each parametric combination was collected. In total, 972 data points were used to generate the parametric sweeps presented in Sec. 6.

Table 1: Numerical specimen DEM microproperties

Micro parameter	Value (DEM)
E_i	1 MPa
k_s/k_n	0.5
ϕ_b	30^o
γ_{int}	1.329
Sphere radius	unif(0.75 mm,1.25 mm)
Sphere density	2600 kg/m^3

6. Results and Discussion

Results show how the combined acceleration techniques of matrix factor reuse, multi-threaded factorization, GPU accelerated factorization, and parallel task management im-

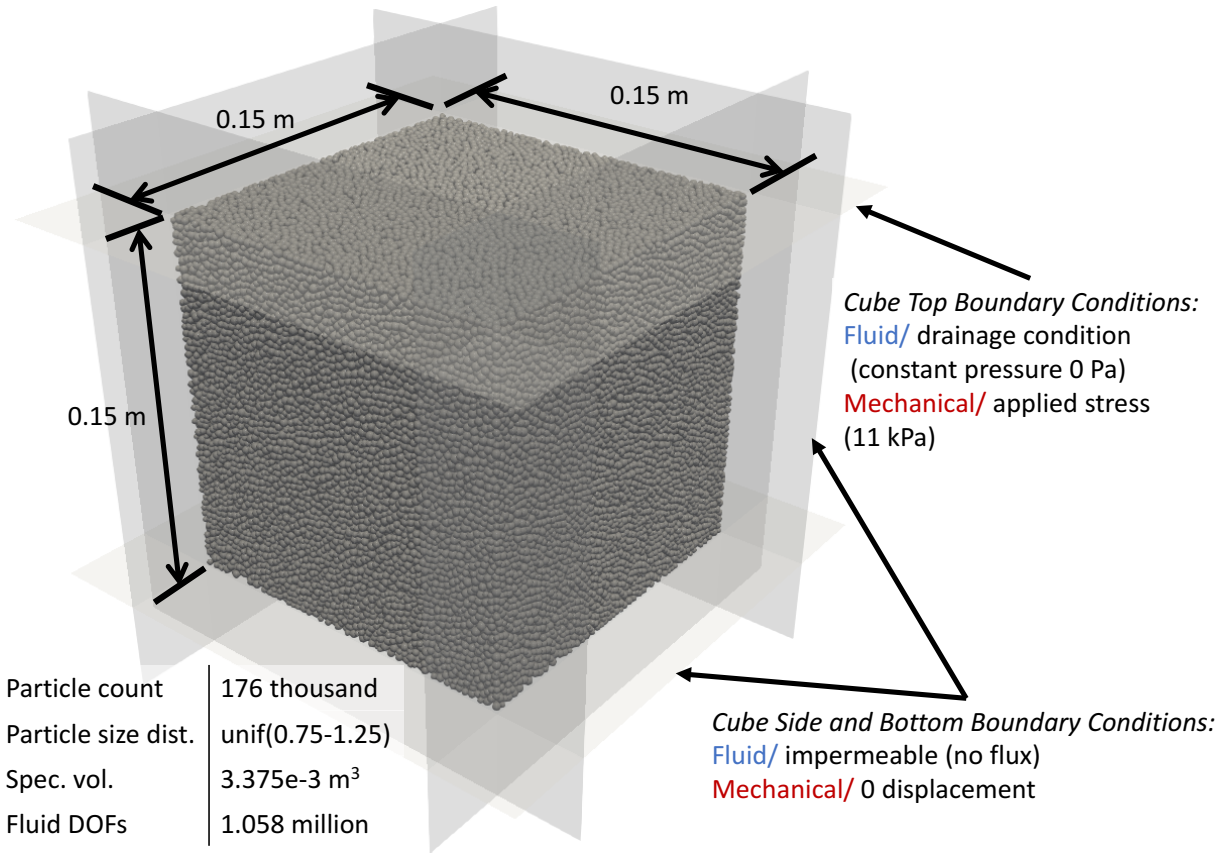

Figure 4: Example of one of the cubical DEM+PFV 1-D consolidation models used to test performance of GPU accelerated factorization.

prove performance by 170x (Figure 5), enabling continuous simulation of poroelastic problems reaching 2.1 million DOFs on an office workstation. The first acceleration technique, **matrix factor reuse**, has the greatest impact on performance by reducing the frequency of rebuilding, reanalyzing, and refactorizing the conductivity matrix (10) according to the selected remesh interval. Results show that these operations consume up to 140 seconds for a system with 2.1 million DOFs (Figures 6 and 7). Without matrix factor reuse, these expensive operations are performed every iteration despite only being necessary after large deformations(Eq. 14) (i.e. matrix factor reuse acceleration is proportional to the selected remesh interval and the dynamics of the system). The second acceleration technique, **GPU accelerated factorization**, decreases the conductivity matrix factorization (Eq. 12) time by 75% compared to a 10-core CPU for 2.1 million DOFs (Figure 6). However, the total t_{bg} is only decreased by 50% due to the single-threaded analyze step comprised of matrix graph partitioning and preconditioning. Although most simulations require the costly analyze step, certain stiff poroelastic simulations benefit from its elimination since it simply reorders and prepares the matrix for factorization. For example, the Discrete Fracture Network model in Yade benefits from reusing the matrix reordering for subsequent factorizations since the non-zero pattern remains constant. Both matrix factor reuse and GPU accelerated factorization contribute to significant gains in performance for the poroelastic oedometer simulation, while the third technique, **multithreaded factorization**, removes the computational time associated with the conductivity matrix factorization by parallelizing the operations with the primary DEM simulation. Therefore, multithreaded factorization increases the optimal remesh frequency associated with an uninterrupted simulation (i.e. conductivity matrix factorization occurs in less time than the time required for the primary simulation to step through one remesh interval Eq. 15). As shown in Figure 8 the time spent per iteration is almost identical for all three devices, which means that the factorization is fully backgrounded in these oedometer simulations. However, it is worth noting that the poroelastic simulation runs 10% faster when the GPU participates, suggesting CPU resources are less

strained when the burden of factorization is taken by the GPU. In any case, the optimal remesh interval and Cundall numbers, show how the GPU is only beneficial for cubical packings $\geq$ 30 thousand particles. Larger cubical packings comprised of $\geq$ 30k particles allow the Tesla K20 to improve λ_{rm} by up to 42%, which means the Tesla K20 enables the update of [**G**] almost two times more frequently than the 1050 Ti for cubical packings comprised of 356 thousand particles. Fig. 8 also shows how the GPU increases the Cundall number by up to 12% for 180k particle packing. Meanwhile, for small cubical packings comprised of $\leq$30 thousand particles, the time spent moving information to and from the GPU outweighs the time saved by the accelerated GPU factorization, resulting in less favorable λ_{rm} and Cundall numbers compared to the 10 core CPU. The final acceleration technique, **parallel task management**, accelerates the coupled solution by ca. 1.2x (Fig. 9). A closer look shows how the time spent on these parallelized algorithms (solving for pore pressures, computing pore volumes, and computing fluid forces) is nearly equivalent to the time spent running the full DEM interaction loop for large particle packings (Figure 8 and 10). The result is misleading since it implies that a coupled simulation should run at exactly the same speed as an uncoupled simulation (provided equivalent core counts). In fact, auxiliary tests show that uncoupled DEM tests run approximately 1 order of magnitude faster than the parallelized coupled DEM+PFV simulation. Ultimately, the CPU L2 and L3 cache sizes in addition to the RAM speed likely limit the linear scalability of increased instruction requests for the coupled simulation in the highly parallelized environment. The point is supported by Figure 10, showing that requesting CPU resources for factorization in addition to foreground FlowEngine algorithms slows the simulation down by 17%. Instead, the GPU factorization technique frees up CPU resources for computing foreground FlowEngine algorithms. Finally, the acceleration benefit for parallel task management decreases as the the system sizes increase and the time spent factorization the matrix dominates the total simulation time (Fig. 9).

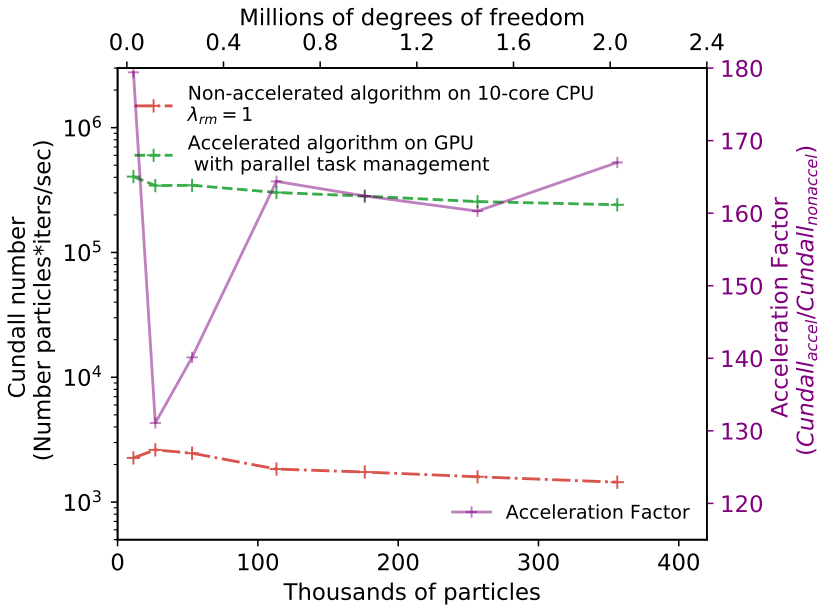

Figure 5: Performance comparison for non-accelerated and fully accelerated algorithms.

7. Summary

Yade's poromechanically coupled DEM+PFV scheme was accelerated by 170x by combining four techniques: matrix factor reuse, multithreaded factorization, GPU accelerated factorization, and parallel task management. Each technique ameliorated different weaknesses associated with the time-dependent implicit pore finite volume scheme. First, **matrix factor reuse** has the largest impact of on performance by reducing the frequency of the costly conductivity matrix factorization. Second, **multithreaded factorization** parallelizes the costly factorization in the background while the coupled simulation steps through time, thus reducing computational cost and enabling an increase of factorization frequency without additional computational time. Third, **GPU accelerated factorization** moves the computational cost of matrix factorization to a Tesla K20 GPU where it factorizes the matrix 75% faster for large poroelastic problems, thus further increasing the factorization frequency by 42% without adding computational time. Finally, **parallel task management** accelerates the solution by 30% by parallelizing auxiliary PFV algorithms (e.g. volume calculations and force calculations) with the DEM interaction loop. All techniques combined

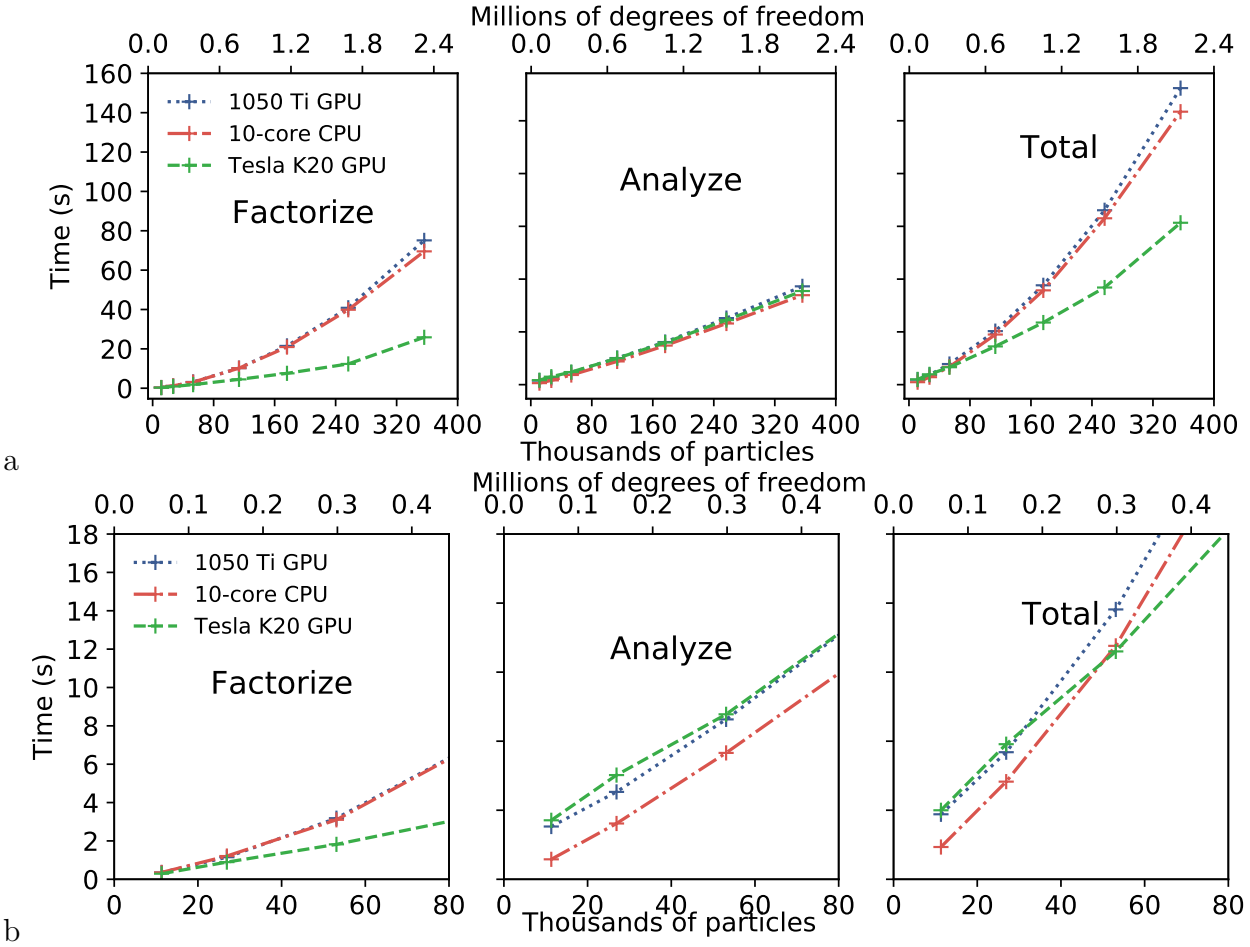

Figure 6: a) Time required to factorize and analyze the conductivity matrix (Eq. 10). $t_{bg} = t_{factor} + t_{analyze}$ b) Zoomed in to show devices timings for small packings (bottom)

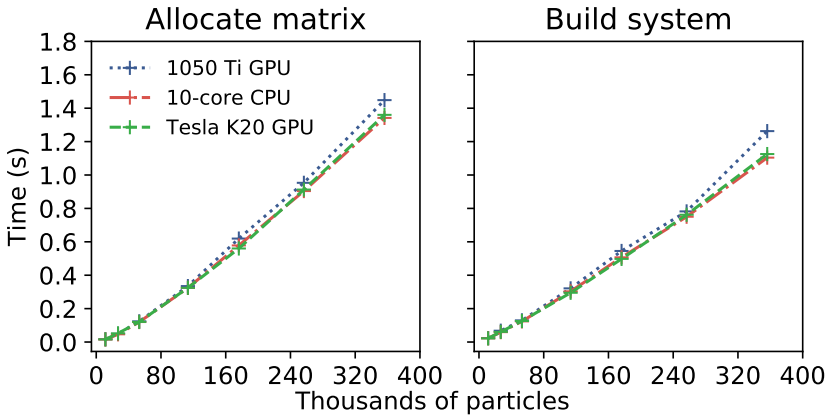

Figure 7: Time required to allocate conductivity matrix (Eq. 10) to memory (left) and build the system of equations (Eq. 10) (right)

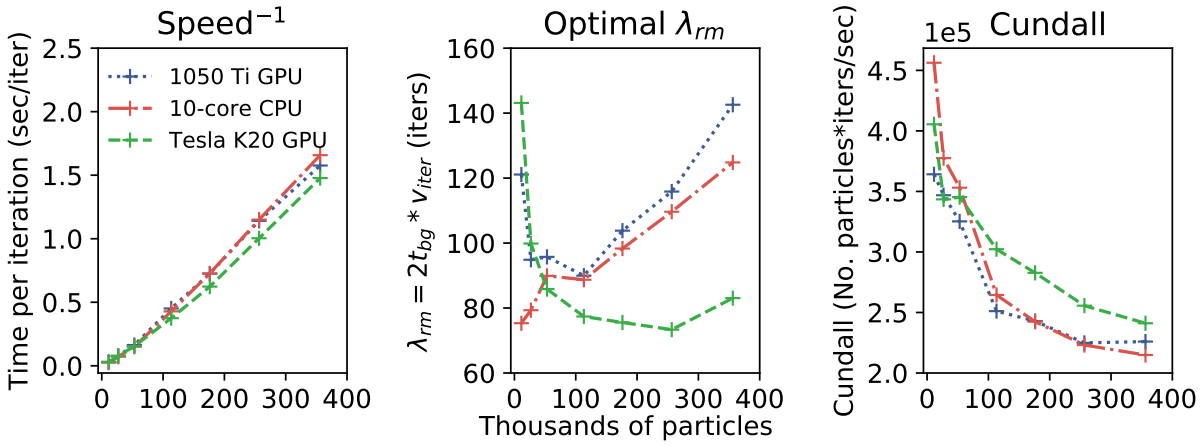

Figure 8: Time per iteration, optimal remesh interval (λ_{rm}) associated with v_{iter} and t_{bg}, and Cundall number for various conductivity matrix (Eq. 10) sizes

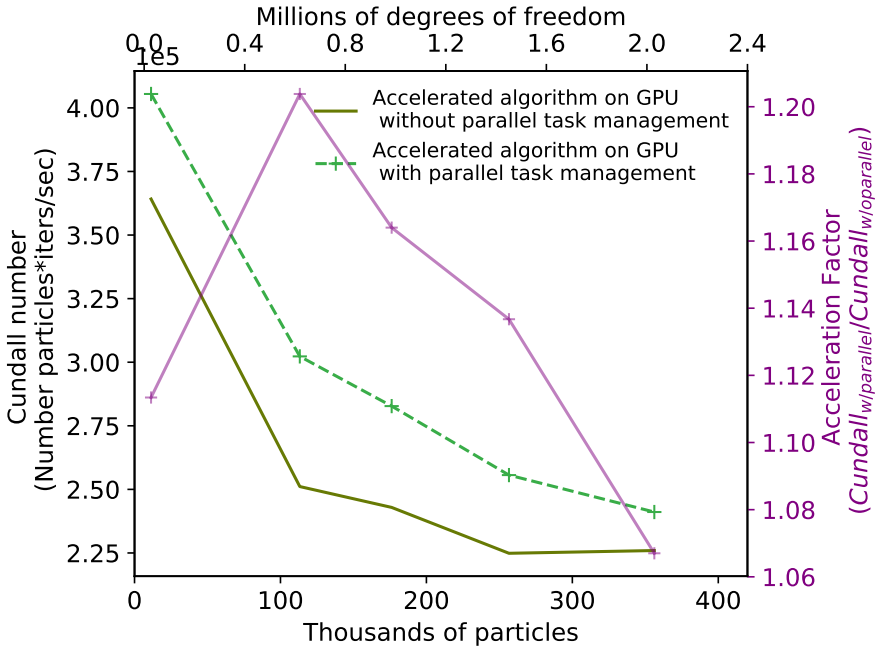

Figure 9: Performance gain from implementing OpenMP parallel task management (Fig. 3).

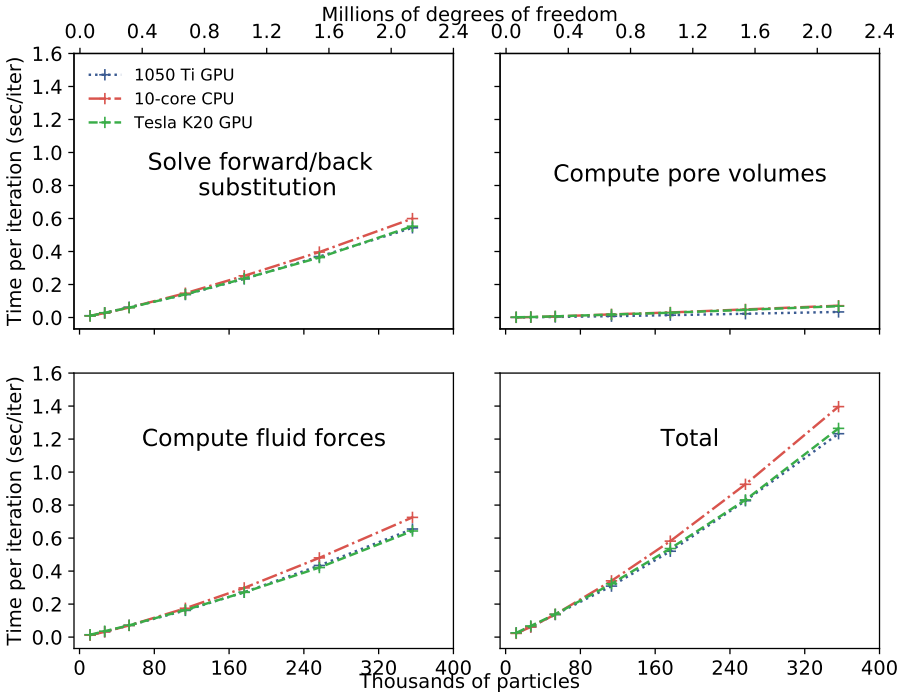

Figure 10: Wall time spent solving for pressures (Eq. 12), computing pore volumes, and computing forces (Eq.13) (parallel task management in Fig. 3).

enable the simulation of poroelastic systems comprised of of 2.1 million DOFs (356 thousand particles) on an office workstation. After reducing the cost of factorization, the new limitation lies in CHOLMOD's analysis step, which orders the matrix and builds the elimination tree on a single thread. However, specialized stiff fracture network simulations can avoid this step entirely by reusing the matrix ordering for subsequent factorizations. Future improvements will focus on the parallelization of the analysis step, MPI solutions for systems with 10s of millions of DOFs, and updating/downdating matrix factorizations depending on the magnitude of rank change.

8. Acknowledgments

This work was funded partially by Laboratoire 3SR of Univ. Grenoble Alpes.

References

Booth, J. D., Chatterjee, A., Raghavan, P., and Frasca, M. (2011). A multilevel Cholesky conjugate gradients hybrid solver for linear systems with multiple right-hand sides. *Procedia Computer Science*, 4:2307–2316.

Camborde, F., Mariotti, C., and Donzé, F. V. (2000). Numerical study of rock and concrete behaviour by discrete element modelling. *Computers and Geotechnics*, 27(4):225–247.

Catalano, E. (2012). *A pore-scale coupled hydromechanical model for biphasic granular media. Application to granular sediment hydrodynamics*. PhD thesis, Universite de Grenoble.

Catalano, E., Chareyre, B., and Barthelemy, E. (2014). Pore-scale modeling of fluid-particles interaction and emerging poromechanical effects. *International Journal for Numerical and Analytical Methods in Geomechanics*, 38(1):51–71.

Catalano, E., Chareyre, B., Cortis, A., and Barthelemy, E. (2011). A pore-scale hydromechanical coupled model for geomaterials. In *II International Conference on Particle-based Methods. Fundamentals and Applications*, number January 2011, pages 798–809.

Chalak, C., Chareyre, B., Nikooee, E., and Darve, F. (2017). Partially saturated media: from DEM simulation to thermodynamic interpretation. *European Journal of Environmental and Civil Engineering*, 21(7-8):798–820.

Chareyre, B., Cortis, A., Catalano, E., and Barthélemy, E. (2012). Pore-Scale Modeling of Viscous Flow and Induced Forces in Dense Sphere Packings. *Transport in Porous Media*, 94(2):595–615.

Chen, K. (2005). *Matrix preconditioning techniques and applications*, volume 19. Cambridge University Press.

Cundall, P. A. and Strack, O. D. L. (1979). A discrete numerical model for granular assemblies. *Géotechnique*, 29(1):47–65.

Davis, T. a. (2013). User Guide for CHOLMOD : a sparse Cholesky factorization and modification package. *Department of Computer and Information Science and ...*, pages 1–140.

Davis, T. A. and Hager, W. W. (2009). Dynamic supernodes in sparse cholesky update/-downdate and triangular solves. *ACM Transactions on Mathematical Software*, 35(4).

Detournay, E. and Cheng, A. H. (1993). Fundamentals of poroelasticity. *Comprehensive rock engineering. Vol. 2*.

Edwards, D. A., Shapiro, M., Brenner, H., and Shapira, M. (1991). Dispersion of inert solutes in spatially periodic, two-dimensional model porous media. *Transport in Porous Media*, 6(4):337–358.

Hosn, R. A., Sibille, L., Benahmed, N., and Chareyre, B. (2017). Discrete numerical modeling of loose soil with spherical particles and interparticle rolling friction. *Granular matter*, 19(1):4.

Jimack, P. and Touheed, N. (2000). Developing parallel finite element software using mpi. *High Performance Computing for Computational Mechanics*, pages 15–38.

Ju, S.-H. (2010). A simple openmp scheme for parallel iteration solvers in finite element analysis. *Computer Modeling in Engineering & Sciences(CMES)*, 64(1):91–108.

Kakay, A., Westphal, E., and Hertel, R. (2010). Speedup of fem micromagnetic simulations with graphical processing units. *IEEE transactions on magnetics*, 46(6):2303–2306.

Liu, Y., Jiao, S., Wu, W., and De, S. (2008). Gpu accelerated fast fem deformation simulation. In *APCCAS 2008-2008 IEEE Asia Pacific Conference on Circuits and Systems*, pages 606–609. IEEE.

O'Sullivan, C. (2011). *Particulate discrete element modelling: a geomechanics perspective*. CRC Press.

Papachristos, E., Scholtès, L., Donzé, F., and Chareyre, B. (2017). Intensity and volumetric characterizations of hydraulically driven fractures by hydro-mechanical simulations. *International Journal of Rock Mechanics and Mining Sciences*, 93:163–178.

Rennich, S. C., Stosic, D., and Davis, T. A. (2016). Accelerating sparse Cholesky factorization on GPUs. *Parallel Computing*, 59:140–150.

Scholtès, L. and Donzé, F. V. (2012). A DEM model for soft and hard rocks: Role of grain interlocking on strength. *Journal of the Mechanics and Physics of Solids*, 61:352–369.

Šmilauer, V. and Chareyre, B. (2015). DEM Formulation, Release Yade documentation. *Yade Documentation*, pages 137–160.

Smith, I. M., Griffiths, D. V., and Margetts, L. (2013). *Programming the finite element method*. John Wiley & Sons.

Sweijen, T., Hassanizadeh, S. M., Chareyre, B., Zhuang, L., Sweijen, T., Hassanizadeh, S. M., and Thompson, K. E. (2018). Dynamic pore-scale model of drainage in granular porous media: the pore-unit assembly method. *Water Resources Research*.

Wang, H. F. (2017). *Theory of linear poroelasticity with applications to geomechanics and hydrology*. Princeton University Press.

Weatherley, D., Boros, V., and Hancock, W. (2011). Esys-particle tutorial and users guide version 2.1. *Earth Systems Science Computational Centre, The University of Queensland*.

Willingham, T. W., Werth, C. J., and Valocchi, A. J. (2008). Evaluation of the effects of porous media structure on mixing-controlled reactions using pore-scale modeling and micromodel experiments. *Environmental Science and Technology*, 42(9):3185–3193.

Yuan, C., Chareyre, B., and Darve, F. (2017). Deformation and stresses upon drainage of an idealized granular material.

Zhu, H. P., Zhou, Z. Y., Yang, R. Y., and Yu, A. B. (2008). Discrete particle simulation of particulate systems: A review of major applications and findings. *Chemical Engineering Science*, 63(23):5728–5770.

Zoback, M. D. (2007). *Reservoir Geomechanics*. Cambridge University Press.

Appendix A. Stability of the coupled algorithm

In coupled simulations, the fluid surrounding particles acts as a viscous damper, which results in a force to be added to the contact forces. After substituting the drag forces with equations 10 and 13 the Newton's second law of motion can be written, formally:

$$\mathbf{M}\ddot{\mathbf{x}}^{[t]} + \mathbf{V}^{[t]}\dot{\mathbf{x}}^{[t]} + \mathbf{K}^{[t]}\mathbf{x}^{[t]} = 0 \tag{A.1}$$

where $\mathbf{x}$ is the generalized particle position, $\mathbf{M}$ and $\mathbf{K}$ express the global mass and stiffness matrices, and the viscous matrix is comprised of the inverted conductivity matrix, the global force matrix and the global volume rate matrix, $\mathbf{V} = \mathbf{F}\mathbf{G}^{-1}\mathbf{E}$. The stability of the explicit time integration scheme for this equation is now discussed by considering two limit cases: stiffness dominated regimes and viscosity dominated regimes.

In stiffness dominated systems, the stability of the oscillating spring-mass system is simply a function of the natural period of the system (see the appendix of Hosn et al. (2017) for a detailed derivation):

$$\Delta t_{M-K} = \min\left(\sqrt{m_k/K_{k,i}}\right) \tag{A.2}$$

where m_k and $K_{k,i}$ are particle k mass and equivalent stiffness (considering all particle k contacts and degrees of freedom, i).

If stiffness effects are negligible compared to viscous effects, we derive the stability criteria as follows:

$$\ddot{\mathbf{x}}^{[t]} + \mathbf{V}^{[t]}\mathbf{M}^{-1}\dot{\mathbf{x}}^{[t]} = 0 \tag{A.3}$$

$$\frac{\dot{\mathbf{x}}^{[t+\Delta t]} - \dot{\mathbf{x}}^{[t]}}{\Delta t} + \mathbf{V}^{[t]}\mathbf{M}^{-1}\dot{\mathbf{x}}^{[t]} = 0 \tag{A.4}$$

$$\dot{\mathbf{x}}^{[t+\Delta t]} = \left(\mathbf{I} - \mathbf{V}^{[t]}\mathbf{M}^{-1}\Delta t\right)\dot{\mathbf{x}}^{[t]} \tag{A.5}$$

let $\mathbf{V}\mathbf{M}^{-1} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ be the eigenvalue decomposition of $\mathbf{V}\mathbf{M}^{-1}$, where $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues and $\mathbf{U}$ is the matrix of eigenvectors. Plugging $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ into Eq. A.5 yields:

$$\mathbf{U}^{-1}\dot{\mathbf{x}}^{[t+\Delta t]} = \mathbf{U}^{-1}\dot{\mathbf{x}}^{[t]} - \Delta t\mathbf{\Lambda}\mathbf{U}^{-1}\dot{\mathbf{x}}^{[t]} \tag{A.6}$$

and the transformation between between coordinates is denoted as $\dot{\mathbf{y}}^{[t]} := \mathbf{U}^{-1}\dot{\mathbf{x}}^{[t]}$:

$$\dot{\mathbf{y}}^{[t+\Delta t]} = (\mathbf{I} - \Delta t\mathbf{\Lambda})\dot{\mathbf{y}}^{[t]} \tag{A.7}$$

which is a set of scalar equations since $\mathbf{\Lambda}$ is diagonal. Thus, stability is ensured by imposing:

$$|1 - \Delta t\lambda_{max}| < 1 \tag{A.8}$$

where λ_{max} is the maximum eigenvalue of the matrix $\mathbf{V}^{[t]}\mathbf{M}^{-1}$. Finally, the viscous dominated timestep is computed as:

$$\Delta t_{M-V} = \Delta t < 2 \cdot \lambda_{max}^{-1} \tag{A.9}$$

In an attempt to relieve the computational expense associated with determining the eigenvalues of $\mathbf{V}^{[t]}\mathbf{M}^{-1}$ (which would need to invert $\mathbf{G}$), a parametric analysis was performed to investigate the distribution of viscous coefficients for polydispersed granular packings. In brief, a non-zero velocity was imposed on each particle and the resulting viscous force was measured. An empirical upper bound of $m_k/\mathbf{v}_k^{[t]}$ in dense packings was found as:

$$\frac{m_k}{\mathbf{v}_k^{[t]}} < \frac{1}{8000}\frac{\pi\rho_k\phi_k^2}{\mu} \tag{A.10}$$

where ρ_k, ϕ_k, and μ are particle k density, particle k diameter, and fluid viscosity. Considering

$\min(m_k/\mathbf{v}_k^{[t]}) \approx \lambda_{max}^{-1}$, the empirical estimate for $m_k/\mathbf{v}_k^{[t]}$ is inserted into Eq. A.9 to yield a fast estimate of the maximum viscous timestep as:

$$\Delta t_{M-V} < 2 \cdot \min\left(\frac{1}{8000}\frac{\pi\rho_k\phi_k^2}{\mu}\right) \tag{A.11}$$

The final maximum allowed timestep for the coupled scheme in viscous or stiffness dominated regimes is as follows, where the 0.8 pre-factor is enough to ensure stability even in mixed elastic-viscous regimes:

$$\Delta t = 0.8\min(\Delta t_{M-V}, \Delta t_{M-K}) \tag{A.12}$$