EPJ Web of Conferences **249**, 09006 (2021)
Powders and Grains 2021
https://doi.org/10.1051/epjconf/202124909006

Grid-algorithm improvements for dense suspensions of discrete element particles in finite element fluid simulations

Jan Mueller[1,*], *Akira* Kyotani[1], and *Hans-Georg* Matuttis[1,**]

[1]Dep. of Mech. Eng. and Intelligent Systems, The University of Electro-Communications, 182-8585, 1-5-1 Chofugaoka, Chofu, Tokyo

Abstract. For homogeneous systems like classical fluid dynamics and structural mechanics, finite element method (FEM) grid generation has reached a mature state. On the other hand, for multi-physics-problems like fluids with a high density of immersed particles, many researchers may not even be aware of the types of instabilities which may be triggered by unsuitable meshes. We review common types of grid generation, point out previously unrecognised types of instabilities for particles in fluids as well as remedies to obtain particle-fluid simulations with higher stability and fewer redundant degrees of freedom.

1 Introduction

While the simulation of particles in fluids is of practical relevance, both the algorithmic complexity as well as the computational effort are rather forbidding. Many issues are related to the grid generation, which is easy for the traditional fluid mechanics setting of one or few fixed obstacles, but may lead to unforeseen problems for many moving particles which may come arbitrarily close. Outlining the problems and solutions for generating simulation grids with high particle densities is the purpose of this work.
Elongated DEM-particles are needed for granular assemblies with realistic stress-strain relations and angles of repose [1]. In the following, we use polygonal particles, because they allow more stable force computations and we don't need to deal with curved surfaces. Details about the simulation method can be found in [2].

Common simulation methods for particles in fluids are finite difference methods (FD), finite volume methods (FV), finite element methods (FEM) as well as particle methods. We focus on FEM methods, because with unstructured triangular grids they can exactly resolve the boundary conditions of arbitrary polygonal particle packings. Finite difference methods need square grids, which give zero-order approximations for polygons, and finite volume methods need the same connectivity over the whole computational domain, which may be difficult to realize for a pore space between particles of arbitrary shape and size. On the other hand, particle methods for fluids have problems with shot noise and resolution.

To simulate fluid-solid interaction in FEM, the three types of boundary conditions in Fig. 1 are commonly used: In "macroscopic simulations" like in Höfler et al. [3], the flow goes through the particles and forces between fluid and particles are introduced ad hoc. For "immersed boundary conditions", slip is allowed between particle surfaces and the contacting fluid, but at least volume exclusion is retained. We deal exclusively with a "microscopic simulation" with a no-slip condition for the fluid on the particle boundaries, as this is the only truly physical boundary condition between particles and incompressible flow.

For the code, wall correction factors for sinking particles [2] as well as Strouhal numbers and drag coefficients for flows around circular and square cylinders were verified; the deviation from the reference values was insignificant. The effective viscosity was verified for an older version with second order elements and inferior grid generation for monodisperse particles: The agreement with the linear prediction of the "Einstein formula" for two dimensions was excellent up to an area fraction of 0.3, where particle collisions and agglomeration became dominant [4].

2 Automatic mesh generation with a background mesh

When obstacles are fixed, as e.g. in common Karman vortex street simulations, the mesh can be constructed around

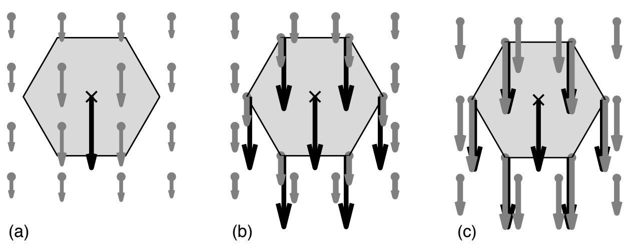

Figure 1. Velocity field of the fluid (grey arrows) around a sinking particle for macroscopic simulation (a), immersed boundary conditions (b) with flow velocity on the particle surface deviating from the particle velocity and microscopic simulation (c) with the particle velocity acting as boundary condition for the fluid.

*e-mail: jan@wilke-it.com
**e-mail: hg@mce.uec.ac.jp
A video is available at https://doi.org/10.48448/y7ry-em34

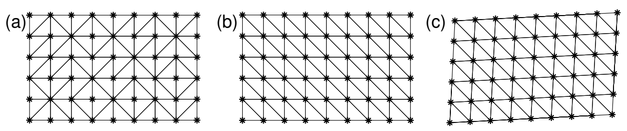

Figure 2. Delaunay-triangulation of a square grid with random connections due to spurious rounding errors from floating-point positions in (a) and rectified Delaunay-mesh in (b) obtained by deforming the mesh as shown in (c).

the particle. However, when dealing with multiple moving particles, the grid-structure must be adapted to the changing geometry every timestep. An obvious approach to automate mesh generation in this case is what we shall call "background mesh": Inside the fluid domain (without the particles), mesh points are set up regularly (e.g. on a square grid). Afterwards the particles are superimposed and mesh nodes near or overlapping the particles are locally rearranged to accomodate the new geometry, while the overall mesh structure is mostly kept intact [5].

2.1 Delaunay-grid

One of the simplest approaches to obtain a triangular mesh is the (constrained) Delaunay-construction. From a square grid of points, Delaunay-triangulation produces the conventional "FEM mesh" or "Friedrichs-Keller grid" (see Fig. 2). It is a structured grid, i.e. all grid points have the same connectivity (except on boundaries). When particles move over this grid, the mesh connectivity is rearranged to take into account the particle outlines. For polygons, the Delaunay-triangulation is run over the background mesh points which are not occluded by the particle, together with the points of the particle outlines [6]. Any triangulation inside the particles is forbidden. (Such a "constrained Delaunay-triangulation" may e.g. violate the uniqueness of the "pure" Delaunay-triangulation.) As the particles disrupt the original connectivities of the pure Friedrichs-Keller grid, the original grid structure gets lost.

2.2 Mesh-relaxation

The equations for the FEM-solution step have the best condition (lowest condition number) if the triangles are equilateral. As a rule of thumb, FEM-triangles should not contain angles above 135 degrees [7]. However, a Friedrichs-Keller background mesh combined with the arbitrary orientation of poligonal particles will also cause arbitrary angles in the mesh elements. As a remedy, we introduced a relaxation of the grid [6]: Edges between grid points $r_n^{\cdots}$ (the $^{\cdots}$ are dropped in the following) are treated as springs, with an equilibrium position for endpoints as the corners of equilateral triangles. The Störmer-Verlet method

$$r_{n+1} = 2r_n - r_{n-1} + \iota^2 a_n \tag{1}$$

of second order is rewritten with an "iteration time step" ι (wich is virtual time and not related to the physical timestep τ) and the acceleration a_n (representing the spring force) as

$$r_{n+1} = r_n + \underbrace{\frac{r_n - r_{n-1}}{\iota}}_{v_n} \iota + \iota^2 a_n \tag{2}$$

and drop the term v_n, which introduces an error of first order, so that the accuracy has order zero. Thus, the integrator can only find equilibrium positions.

However, we observed the relaxation algorithm as described in [6] is prone to squishing/flipping mesh elements or pushing nodes into particles. This occurs more often in geometries with mesh elements of very different sizes, either due to the current arrangement of the particles or the use of graded mesh (see Sec. 2.3). While this can be avoided by using only a limited amount of relaxation iterations and by resetting the backround mesh to its initial state at regular physical timesteps, this is not a reliable foundation for meshing increasingly complex geometries (i.e. more particles). Instead we introduced two methods to protect small mesh elements from excessive deformation: First, using the local mesh size of each element for the calculation of the spring forces a_n instead of a global average, reduces overall tension in regions of fine mesh that usually triggers flipping of elements. Second, comparing the areas A_n of elements with those of their neighbours $A_{n,i}$ and applying an additional growing/shrinking spring force to all edges (again represented as acceleration)

$$a_{n,gs} = \sum_i \operatorname{sgn}(A_{n,i} - A_n) \cdot \sqrt{|(A_{n,i} - A_n)|} \tag{3}$$

uniformly, allows elements to counter forces of neighbouring elements if they have been squished too much.

2.3 Graded mesh

Very often, the domain of a simulation is much larger than the actual region of interest. In classical fluid dynamics, the simulation of flow around an obstacle with characteristic length D is performed in a domain stretching at least $40D$ downstream and $10D$ in every other direction from the obstacle to avoid disturbances of the actual flow field. For sedimentation simulations, we have to simulate not only the region with particles, but also the surrounding fluid region from where the particles move out. Using

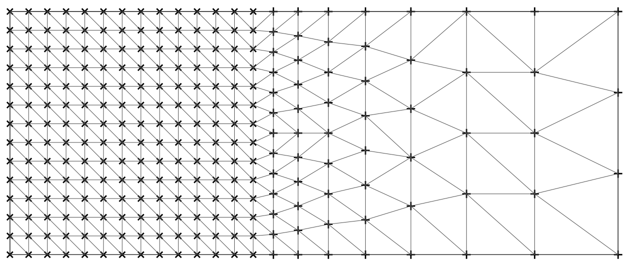

Figure 3. Graded mesh with high resolution in the region of interest (marked with ×) and buffer space towards the boundary (marked with +).

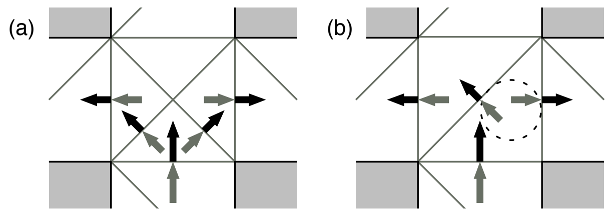

Figure 4. Bifurcating FEM-flow at a junction between four particles (in grey), with grey arrows for the inflow and black arrows for the outflow. Possible configuration with a bifurcation angle at around 90° at the blunt element angle of 90° in (a). Numerically precarious configuration with an outflow twisted by 135° at the sharp element angle of 45° in (b).

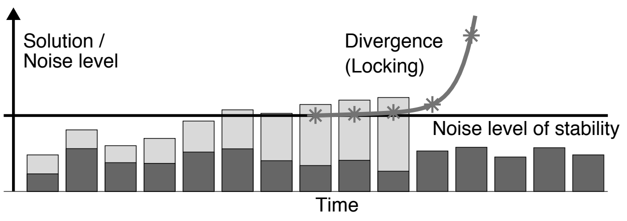

Figure 5. Noise level without (black columns) and with changes in the mesh (light columns) with the divergence (stars) leading to locking.

a uniform mesh size defined by the one around the particles (about one particle edge length) would be a huge waste of computational ressources. Instead, it is preferable to coarsen the grid towards empty boundaries like in Fig.3, where the flow will be slower and has structures of increasingly larger spatial wave length.

3 Locking and instabilities

Usually, convergence problems in the finite element field are summarized as "locking", which for dynamic simulations manifest as an inability to determine the field amplitudes for the next timestep. Locking is predominantly accompanied by a diverging condition number of the FEM-system's Jacobian. Reasons may vary from outright programming mistakes (overlapping elements, non-space-filling grids, misstated boundary conditions) to the degenerate triangles mentioned in the previous section. Another obvious reason for locking is too few or the wrong degrees of freedom (number of nodes) in areas of strong variation, because there are not enough equations to represent smooth changes. Less obvious are too many degrees of freedom for fields with small variations, as equations for neighbouring nodes may become redundant and increase the condition number of the Jacobian. For grids around many particles with non-slip boundary conditions, there are additional mechanisms which lead to locking but are hardly explored in literature and will be explained in the following. The types of instability described can be circumvented by proper mesh generation, but require more steps than just the "background mesh" described in Sec. 2.

3.1 Locking due to wrong element curvature

Eriksson et al. [8] give an example for locking with elements of very high (9th) order. Such examples have led to a belief in the FEM fluid mechanics community that "lower order is better", as 9th order is pretty high even for least squares fits. For flow fields with small curvature and large distances between boundaries, lower (1st or 2nd) order elements may be sufficient, as the curvature of the field within an element remains constant. However, for two close parallel surfaces moving in opposite directions, low order elements cannot model the smooth transition of the velocity from positive to negative direction.This behaviour of finite elements with constant curvature (under which also many finite difference methods can be classified [9]) is a type of instability specific to granular materials in fluids, which has up to now not been noticed in the community. Likewise, for many particles moving arbitrarily and in close proximity inside a fluid, the sharp velocity gradients may lead to locking with low order elements. But mesh refinement in flow channels smaller than an average particle diameter is computationally prohibitive.

We have confirmed that quasi-third order elements $P_2^+P_1$ (second order with cubic "bubble" for velocities and first order for pressure) can deal with the sharp gradients in a velocity field of variable curvature. For fluids alone, larger gradients will be "smeared out" by the viscosity, but regions with particles may lead to higher velocity gradients than those predicted by the Reynolds number.

3.2 Locking due to non-resolved cross-flows

Another reason for locking are junctions between particles where the flow splits or unites. Finite element functions are continuous, so on a single element it will not be algebraically possible to describe splitting of the flow with functions below 4th order. Instead of using the minimum number of mesh triangles, which would force a continuous, non-bifurcated flow, we have to add an additional lattice point to allow the flow to separate into different directions (see Fig. 4). By constructing the Voronoi mesh around the centers of mass of particles, the resulting points can be used as additional nodes if no point of the background mesh is available within the junction.

3.3 Instability due to oscillating grids

Moving particles require a reconfiguration of the mesh in every timestep. The new mesh inherits the old field amplitudes via interpolation, and while the FEM-solution on the unchanged nodes is the same, the amplitudes elsewhere may be modified. Therefore, every interpolation creates noise additional to the inherent discretization error in the system (see Fig. 5). Snapping a relaxed mesh back to the initial background mesh (see Sec. 2.2) causes a jump in the condition number of the FEM Jacobian that needs several physical timesteps to subside. If the geometry causes the relaxation to adjust a region of the mesh (even only two elements) so it oscillates between two states, the interpola-

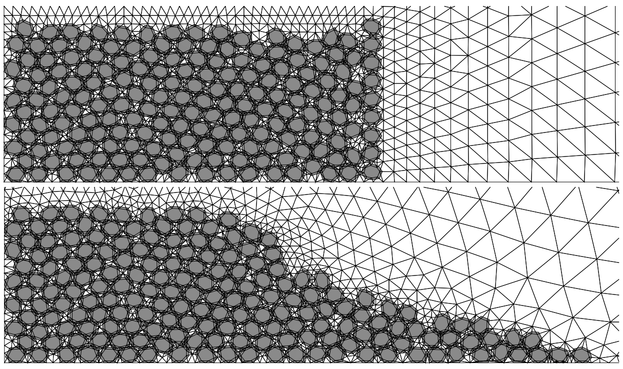
Figure 6. Collapsing granular step. The initially rectangular step is surrounded by a graded mesh (top). After the collapse (bottom) the nodes have slid with the particles to the lower right and vacated areas have integrated with the graded mesh.

tion noise will accumulate. Especially if the oscillation occurs on every timestep, the condition number will rise until the simulation locks. These oscillations can be reduced by limiting the number of mesh relaxation steps per physical timestep, albeit this leads to slightly less equilateral triangles. There are even less oscillations with the "mesh dragging" described in the next section, where the mesh nodes do not snap back to the initial background mesh positions.

4 "Mesh dragging" for more efficiency

In the previous sections, we have introduced the necessary elements to create a suitable mesh for the simulation of particles in fluids. However, setting up a new grid this way for every physical timestep becomes increasingly cumbersome for increasing mesh sizes, particle numbers and densities, so a more efficient approach is needed.

4.1 Unfeasible approaches

Moving only the particles and their boundary points over large distances in an otherwise fixed mesh is not possible. If particles move from areas with fine mesh into areas with coarse mesh, the error in the fluid-solid interaction computation will increase, the triangles may degenerate and the simulation will lock. An approach where every particle carries its own fixed "companion grid" (similar to "chimera grids") may be feasible as long as the distance between particles is high. But for close particles, the algorithmic effort becomes prohibitive: The decision processes for piecing together the companion grids (purging unnecessary points, inserting new Voronoi points, running a relaxation over the seams) has even higher complexity than setting up a new grid every timestep.

4.2 The concept of mesh dragging

Instead of constructing the mesh from scratch for every timestep, it is more effective to construct the grid only once and modify it in the following timesteps. The improved relaxation described in the latter part of Sec. 2.2 is robust enough so it does not collapse small elements and the background mesh does not require to be reset to the initial state. Instead, the particles drag nearby background mesh nodes with them via the relaxation springs in the element edges. We will call this approach "mesh dragging".

This means that a particle entering a coarse mesh region will also tow several mesh nodes with it, keeping the mesh resolution in its neighbourhood roughly constant while the mesh in turn gets coarsened elsewhere (see Fig. 6). Conversely, when a particle enters an area of fine mesh, the relaxation pushes superfluous dragged nodes away from the particle. In this framework, the only mesh generation step required for every timestep is the relaxation. At regular intervals (depending on the mesh size, every 20-50 timesteps turned out to be reliable) the grid is re-triangulated: The constrained Delaunay method updates the neighbourhood relations between grid points, while Voronoi points at flow junctions are recalculated to ensure physical flow directions according to Sec. 3.2.

5 Summary

We have elaborated on three novel types of FEM instabilities which up to now have been unrecognised in the community: Locking caused by elements of constant curvature, by cross-flows between particles and by oscillations of grid-points due to remeshing. The first can be avoided by using third order elements, the second by inserting additional points using Voronoi tesselation and the third by adhering to the mesh generation principles outlined by us. A given "background mesh" can be improved with mesh relaxation and challenges posed by an inhomogenous and variable geometry can further be overcome with the implementation of a graded (background) mesh and "mesh dragging". This allows the construction of grids of reasonable quality with a minimal number of nodes reducing the number of degrees of freedom and thus requiring the least possible amount of computation time.

References

[1] H.G. Matuttis, J. Chen, *Understanding the Discrete Element Method* (Wiley, 2014)
[2] J. Mueller, A. Kyotani, H.G. Matuttis, J. of App. Math. and Phys. **8**, 1779 (2020)
[3] K. Höfler, S. Schwarzer, Phys. Rev. E **61**, 7146 (2000)
[4] T.L. Cheik, Master's thesis, The University of Electro-Communications, Dep. of Mechanical Engineering and Intelligent Systems (2014)
[5] G.H. Ristow, Phys. Rev. E **55**, 2808 (1997)
[6] S.H. Ng, H.G. Matuttis, Theoretical and Applied Mechanics Japan **59**, 323 (2011)
[7] J. van Kan, A. Segal, G. Segal, F. Vermolen, *Numerical Methods in Scientific Computing* (VSSD, 2005)
[8] K. Eriksson, D. Estep, P. Hansbo, C. Johnson, *Computational Differential Equations* (Cambridge University Press, 1996)
[9] P.M. Gresho, R.L. Sani, *Incompressible Flow and the Finite Element Method, Volume 2: Isothermal Laminar Flow* (John Wiley and Sons, Ltd., 2000)