Yade-DEM: (not so) recent advances and steps toward multiphase couplings

B. Chareyre
Univ. Grenoble Alpes, 3SR, F-38000Grenoble, France
CNRS, 3SR, F-38000Grenoble, France
bruno.chareyre[A]3sr-grenoble.fr

ABSTRACT

The paper offers a partial overview of what happened to the Yade-DEM code since the last reports of the “Discrete Element group” of Grenoble (2008). The project has matured considerably. It is now truly international. It has a stable codebase and a large community of users. It provides a rich toolbox for discrete element modeling, applied by researchers in different fields of science. The code has proved to be a robust and flexible foundation for developing innovative numerical methods. It is being extended continuously, in various ways and by different research groups. Some recent developments done at 3SR lab. are outlined, especially those in relation with micro-scale simulations of multiphase problems.

1. INTRODUCTION

Prof. Frédéric Victor Donzé was organizing annual reports of the “Discrete Element Group for Hazard Mitigation” from 2004 to 2008[9]. These reports were good places to discuss not only results obtained with the discrete element method (DEM) but also progresses of the open-source code Yade (dedicated to DEM, and initiated under the supervision F. Donzé). The present annual report is in a sense a descendant of those annual reports, now reflecting the activities of a larger group – for good. It is an ideal opportunity for another progress report regarding Yade. It will partly and quickly fill gaps between 2008 and now (section 2). Some features and perspectives of Yade in the present state will be discussed (section 3). The last part is more focused on the research of the author, it provides a synthetic view of the solid-fluid coupling models developed in the last years (section 4).

2. SHORT HISTORY

The project of an open-source code for DEM has been initiated by F.V. Donzé for two main reasons:

[9] http://people.3sr-grenoble.fr/users/fdonze/Discrete_Element_Group_FVD.html

- an increasing demand for developing (that is, not only “using”) new DEM models, but
- a very limited support locally for developing and maintaining computer codes.

Hence, the idea of stopping Donze's in-house code “SDEC” and to start a more ambitious project which could gather more manpower from different research institutions. Surely, one has to be as optimistic as Frédéric to believe that such an idea has more than a 1% chance of success. Retrospectively, it sounds like a message in a thrown away bottle. Even worst, the actual message in the bottle was “look, my project is ambitious but there is no manpower. Come join us, there is a lot of work and no immediate results expected”. After all, 98% of the open-source projects are dead after one year on Github[10] (not counting those projects which never even reach any public repository). It was very likely to fail. Apparently it did not. There may be rational factors to explain that. Namely, “Open-source projects flourish when developers are also users of the software“[11], yet a good part of luck is not to be excluded.

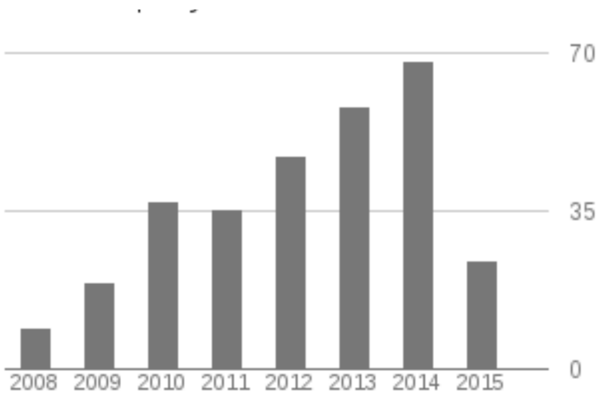

Figure 1. Citations per year in scientific papers[12] (measured 02/2015) .

Lines of code	74,384
Languages	C++ 62%, Python 34%, other 4%
License	GPL-2.0
Commits	4752
Contributors	42
Cost estimate (basic COCOMO Model[13])	18 person-years, $1,006,365

Table 1. Facts and figures (January 2015)

Olivier Galizzi and Janek Koziciki started the project from scratch in 2004. Galizzi commited the first versioned code in january 2005[14] and left the project (and his PhD work) one year

[10]http://redmonk.com/dberkholz/2014/05/02/github-language-trends-and-the-fragmenting-landscape/#ixzz30wEgsUif
[11]http://www.pbs.org/idealab/2013/08/6-things-to-know-about-successful-open-source-software
[12]http://scholar.google.fr/citations?user=hZB8GGcAAAAJ&hl=en
[13]http://en.wikipedia.org/wiki/COCOMO

later. In 2006 some key ideas of the design were already firmly established (as found in Kozicki and Donzé 2009), but Yade was still far from a ready-to-use software. The main deficiencies were reported by Smilauer (2006). Smilauer's paper was in fact announcing the main steps that were effectively taken in the next four years, mainly by Smilauer himself or under his expert guidance (two of them appear in section 3). This evolution came through an almost complete refactoring of the code, which was already quite large and growing at that time. Conducting the infrastructural changes without breaking the superstructure (i.e. mechanical models) needed considerable time and efforts. It did not happen without significant removal (or *cleaning*) of less used or unmaintained parts of the program, occasionally triggering hot debates between the developers. Two years after this big plan had been set up, the deep changes were only starting to reach that part of the iceberg visible to ordinary humans, as suggested by the title of the real last paper about Yade from the Discrete Element Group (Duriez 2008).

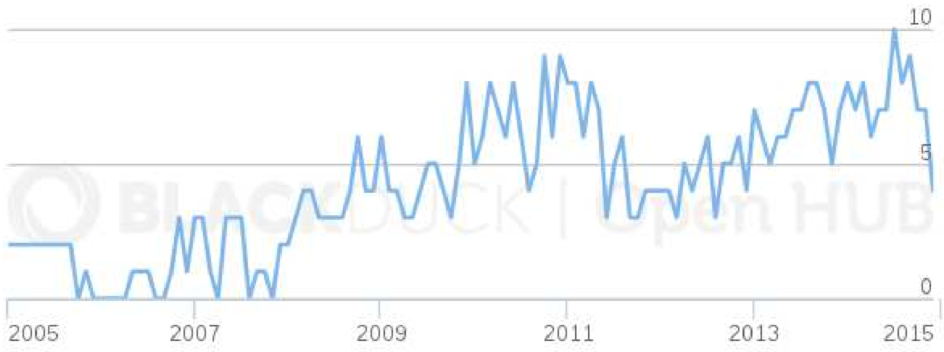

Figure 2. Number of contributors to the source code per month (after BlackDuck OpenHUB[15])

It needed another two years - or almost - to consolidate them and to make them really useful for the average user. In the meantime, another major step had been prepared by Anton Gladky (TU Freiberg): packaging. In other words: integrating the code in standard linux distributions in the form of binary libraries. It enabled running DEM simulations without the need to compile the code. Yade thus became one of those softwares that one finds on the shelf after installing a linux system. It made using Yade even easier, and it helped to keep compatibility with ever-changing operating systems (keeping compatibility is still a serious amount of work, tackled efficiently by Gladky until now).
While all this was happening, researchers[16] where still working on physics models and post-processing methods, enabling progressively the simulation of a large variety of situations. The advances on both sides led a growing audience to adopt Yade for DEM simulations in research. This is assessed by the increasing number of citations per year (fig. 1, mainly cited by papers reporting works done with Yade). The number of developers has been also growing steadily (fig. 2).
In late 2011, Šmilauer started the project Woo-DEM (initially a fork of Yade-DEM) in relation with consulting activities in DEM[17]. The author has been coordinating the Yade project since then.

[14] http://bazaar.launchpad.net/~yade-pkg/yade/git-trunk/files/1
[15] https://www.openhub.net/p/yade
[16] Including the author and students he was supervizing
[17] http://woodem.eu/

3. SOME CHANGES BETWEEN 2009 AND 2015

In this part some important changes since 2009 are commented. More details can be found in the PhD thesis of Šmilauer (2010) or in Yade's documentation (Šmilauer et al. 2010, based on the PhD thesis for a large part).

3.1. BYE BYE I/O FILES, HELLO INTERACTIVE PROGRAMMING

Many computer codes for numerical simulation in physics are based on input data files, first read by a pre-processor or directly by a solver running the mathematical steps. In turns, the solver will write the results in output files (long lists of numbers, typically). Eventually, a post-processor would be used to convert the output files to graphical representations. Yade pre-2010 was not an exception to this good old "I/O files" paradigm (Duriez 2008). The average user had to code a simulation scenario in a C++ class (inheriting form the generic *PreProcessor* class), recompile the code, then run simulations with this pre-processor. It was possible to change some input parameters of the simulation without recompiling (through input files or graphical interfaces), but every pre-processor would run the same scenario forever. A graphical user interface (GUI) was giving the feeling of interactivity, but practically it was only providing to the user the option of time-stepping forward or not (fig. 3). A consequence of this workflow was that one had to be able to write some C++ code and compile it before performing any DEM simulation. This is way above the standard requirements for doing a PhD in physics and mechanics. Moreover - as for every software rooted in the I/O files approach - defining complex situations with different loading steps and conditional transitions between them was tedious if not impossible.

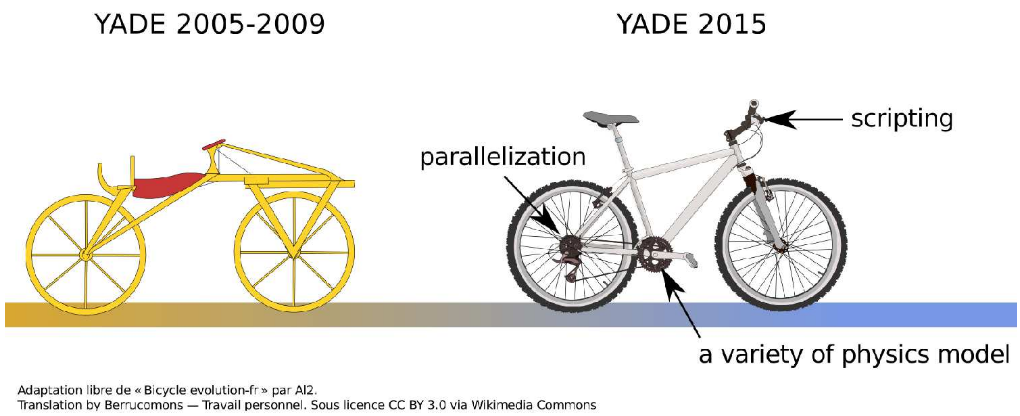

Figure 3. A simplified view of the evolution of Yade over the years.

Integrating the Python language in Yade for commanding and scripting (de facto turning Yade into a Python module) changed this picture completely. It was maybe the most brilliant step taken by Šmilauer. Nowadays, Python is to Yade what Matlab™ is to Comsol Multiphysics™: a powerful programming language with a large set of mathematical and graphical libraries, commanding a more specialized solver interactively. A user can play with particles, add/delete them, change the velocity of an object or a boundary condition, and get an immediate feedback. Graphical representations of the results through dynamic graphs, 3D views or color maps can be generated on the fly. It thus enables true *numerical experiments*, an invaluable method for prototyping numerical models and, simply, for understanding complex mechanical systems. The users need only a basic knowledge in Python programming, which is much less demanding than C++ and does not require compilation[18]. Looking backward gives the feeling that the old PreProcessor class[19] was implying the same level of awkwardness than recompiling Comsol Mutliphysics for solving a different boundary value problem with FEM.

3.2. COUPLING CODES

Interactive programming is not the sole advantage of the Python interface. Another one is that it enables the efficient resolution of coupled problems using partitioned schemes, by combining specialized solvers. Using state-of-the-art solvers developed independently is indeed becoming a method choice in computer simulations. It does not require one to re-implement everything in a single all-singing-all-dancing code (which would result, typically, in sub-optimal versions of every part and unmaintainable framework in the long run with limited manpower). In this context a communication between the codes is required., but the naive approach of communicating through data files is to be avoided. It would be an efficient performance killer (as well as a hard-drive killer). With a higher level command language such as Python, data exchanges can go through live memory easily. Exchanges are even avoided sometimes, when the same memory can be shared by multiple codes. This method can combine programs written in different languages without problems, Fortran, C and C++ typically.Practical examples of code couplings using Yade (fig. 4) include multi-domain and multi-scale DEM-FEM couplings (Stránský and Jirásek 2012, Guo and Zhao 2014), DEM-CFD coupling based on Open Foam (Chen et al. 2011) or in-house fluid model (Maurin et al. 2013), DEM-DNS coupling (Yade-Yales2[20], currently implemented by Deepak Kunhappan[21]). A general coupling framework has been proposed by Jan Stránský (Stránský 2014).

[18] Python scripts are actually compiled, but it is hidden away from the user.
[19] The PreProcessor class is in fact still present in the code and ready to be used, but nobody used it in the recent years.
[20] http://www.coria-cfd.fr/index.php/YALES2
[21] PhD of Univ. Grenoble Alpes, "Numerical modelling of the mechanical behaviour of cellulose fibers in a fluid flow" , work in progress.

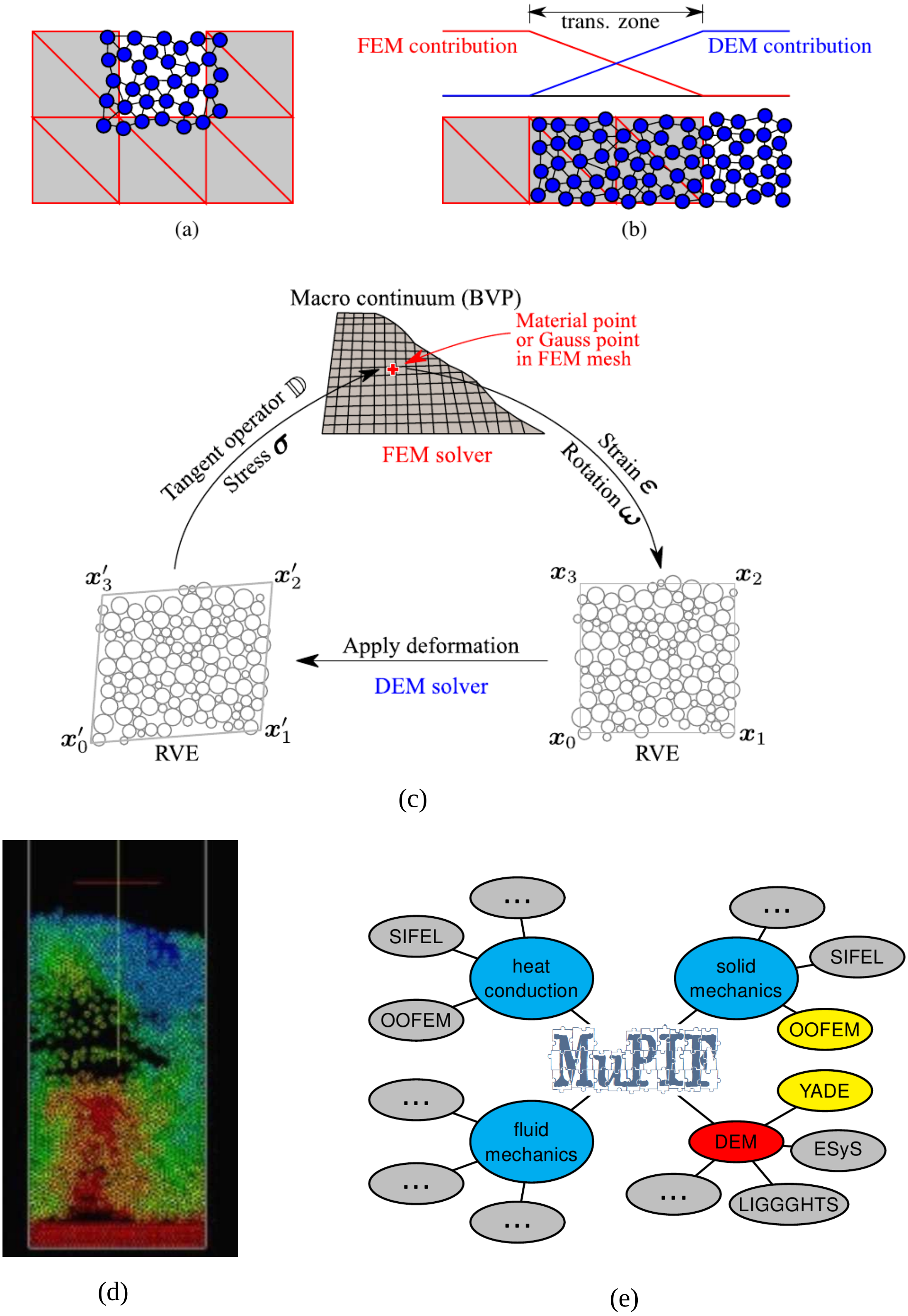

Figure 4. Examples of code couplings. (a) and (b) multidomain FEM-DEM (Stransky 2012), (c) multiscale FEM-DEM (Guo 2014), (d) DEM-CFD (Chen 2011, fluidized bed), (e) a general framework (Stransky 2014).

3.3. PARALLELIZATION

Parallelization and performance have been subjects of continuous brainstorming over the years (Šmilauer 2007, Jakob 2012, Thoeni 2013, Eulitz 2014, Chareyre 2014, Smilauer 2014) even though flexibility has been the primary objective of the design. Šmilauer accomplished the shared memory parallelization of important loops (mainly contacts update and newton integration) using OpenMP. After parallelization of the last non-parallel section (collision detection[22]) by the author, the code was fully OpenMP parallelized. The shared memory approach was, of all possible strategies, the easiest to implement. It can improve the performances by factors up to 7 or 8 for typical large problems on multicore systems (Eulitz 2014). It does not benefit further from large clusters though. This is probably where Yade under-performs some of the other DEM codes[23]. One cure is known: domain decomposition for taking advantage of distributed memory systems (MPI), possibly nesting OpenMP parallelism. Clearly, the lack of manpower is the reason why it did not happen yet. Besides, experiments on many-core computers are still to be performed (Intel's Xeon-Phi co-processors) ; again the man-hours involved in the compilation and the tuning/benchmarking steps is the limiting factor.

4. MULTIPHASE PROBLEMS AT THE MICROSCALE

Numerical models of multiphase granular materials can be obtained by coupling a DEM code with other codes dedicated to fluid dynamics (see section 3.2). This is of course only possible when coupling relatively conventional methods, for which computer codes are available (mainly FEM, CFD, SPH, LBM). When developping less conventional methods and/or couplings, one has – of course – to implement it. The development framework of Yade-DEM has prooved to be an efficient basis for such task. Of course, it provides a ready-to-use DEM library, but it also favors efficient programming through python binding (with a support framework including a set of C++ macro for binding), and inline documentation. Among others, the author pushed developments of this kind, and they are now available as part of Yade.

4.1. ONE PHASE FLOW AT THE PORE SCALE (DEM-PFV)

The pore-scale approach of fluid flow in porous media (the so-called *pore-network* modeling) has been applied to sphere packings and extended to deformable media (Chareyre et al. 2012, Catalano et al. 2014). The numerical scheme – called DEM-PFV – is formaly a special case of ALE methods, where the mesh of the fluid problem follows the movements of the solid particles. Mathematicaly, the problem to be solved is a discrete analog of the coupled equations of poromechanics. Initially written for strictly incompressible fluids, it has been later extended to compressible flow for applications to seabed sediments (Scholtès et al. 2015). A rheological property of saturated materials which does not result simply from the equations of poromechanics is the bulk viscosity. Donia Marzougui showed that this property can be recovered by complementing the poromechanical coupling with short range lubrication forces (Marzougui et al. 2015a/2015b). These developements are done in cooperation with the MEIGE group at lab. LEGI. They are fully integrated in Yade-DEM thus providing a unique

[22] https://yade-dem.org/wiki/Colliders_performace

[23] such as Esys (https://launchpad.net/esys-particle), though no direct benchmarks comparing the two codes are available – unfortunately

tool for applications to various coupled processes (see also Papachristos et al. 2015, and in this volume Toraldo et al. 2015, Tejada et al. 2015, Albaba et al. 2015, Aboul Hosn et al. 2015).

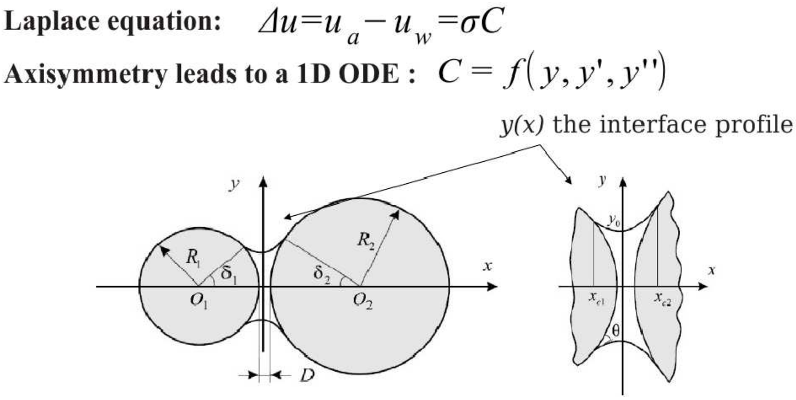

Figure 6. The pendular bridge between two spherical particles.

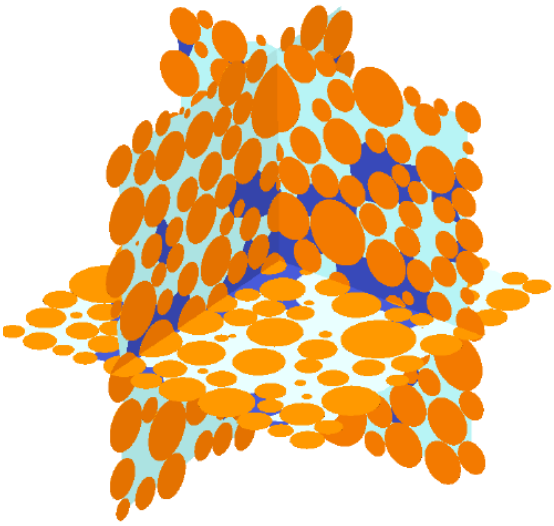

Figure 7. Trapped wetting phase at the end of a simulated drainage (Yuan et al. 2014).

4.2. TWO PHASE PROBLEMS

There has been a number of applications of the DEM to unsaturated materials in the pendular regime in recent years, following Jiang et al. 2004 (2D) and Richefeu et al. 2006 (3D) (see a comparison of various models in Gladkyy and Rüdiger (2014)). The first implementation in Yade is due to Luc Scholtès (Scholtès et al. 2009). An enriched version of this model is being implemented by Caroline Chalak for an accurate determination of interfacial areas (Chalak et al. 2014, Chalak et al. 2015 – in this volume).

The pendular regime is the only case in in which a semi-analytical treatment of Young-Laplace equation is tracktable (fig. 6). In order to approach the full range of saturation, including dynamic regimes in two phase flow, recent developments aims at generalizing the DEM-PFV method for two phases (Yuan et al. 2014, Sweijen et al. 2014, see fig. 7), with local jump conditions between phases during drainage/imbibition events. This a challenging problem, and there is still quite a lot of brainstorming ahead of us. Fortunately, we can count on close cooperations with acknowledged experts at Univ. of Utrecht (Majid S. Hassanizadeh and Ehsan Nikooee / Hydrology Group), advanced experimental techniques there and locally at 3SR (Kaddhour et al. 2013), and through Yade the perfect numerical platform for implementation and dissemination. No excuse to not progress.

ACKNOWLEDGEMENTS

The author wants to thank past and present developers of Yade-DEM for their invaluable contributions, enthusiasm, and professional commitment to provide an efficient tool for DEMing in research and industry, and the Yade community as a whole for feedback, questions, and discussions on the mailing lists yade-users@lists.launchpad.net and yade-dev@lists.launchpad.net. Project hosting by Launchpad and GitHub is greatly appreciated, as well as technical support by Rémi Cailletaud (3SR, Grenoble). Let us not forget the developers of the many third-party libraries used by Yade.
The author wants to acknowledge the support of Grenoble INP for starting the research on hydromechanical coupling (BQR 2008 – PhD of E. Catalano), of the ministry of Ecology Sustainable Development and Energy through project “Hydrofond” (C2D2 program / RGCU), and of the ITN Mumolade (EC FP7).

REFERENCES

Aboul Hosn, R. Sibille, L. Chareyre, B. Benhamed N. (2015), Description of the mechanical degradation of soils subjected to an internal erosion process by suffusion, Geomechanics group annual report.

Albaba, A. Lambert, S. Nicot, F. Chareyre, B. (2015), DEM simulation of dry granular flow impacting a rigid wall, Geomechanics group annual report.

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

Chalak, C., Chareyre, B., Nikooee, E., Hassanizadeh, M., & Darve, F. (2014) DEM Simulations of Unsaturated Soils Interpreted in Thermodynamic Framework. In *ALERT Geomaterials Workshop*, p.11.

Chalak, C., Chareyre, B., Darve, F. (2015), DEM simulations of unsaturated soils interpreted in a thermodynamic framework., Geomechanics group annual report.

Chareyre, B. (2014). What happened to the insertion sort collider? *in Booklet of the 1st YADE Workshop*, Grenoble, 2014.

Chareyre, B., Cortis, A., Catalano, E., Barthélemy, E. (2012), Pore-scale modeling of viscous flow and induced forces in dense sphere packings. Transport in Porous Media (92), pages 473-493.

Chen, F., Drumm, E., Guiochon G. (2011), Coupled discrete element and finite volume solution of two classical soil mechanics problems. Computers and Geotechnics. DOI 10.1016/j.compgeo.2011.03.009

Duriez, J. (2008), Yade for the non-geeks, in Annual Report of Discrete Element Group for Hazard Mitigation.

Eulitz, A. (2014). Performance benchmarking in Yade. *in Booklet of the 1st YADE Workshop*, Grenoble, 2014.

Gladkyy, A., Schwarze, R. (2014), Comparison of different capillary bridge models for application in the discrete element method. Granular Matter, 16(6), 911-920.

Guo, N., & Zhao, J. (2014). A coupled FEM/DEM approach for hierarchical multiscale modelling of granular media. *International Journal for Numerical Methods in Engineering*, *99*(11), 789-818.

Jakob, C. (2012). *Comparison with PFC3D*, online content on the Yade-DEM website. https://yade-dem.org/wiki/Comparisons_with_PFC3D

Jiang, M. J., Leroueil, S., & Konrad, J. M. (2004). Insight into shear strength functions of unsaturated granulates by DEM analyses. Computers and Geotechnics, 31(6), 473-489.

Kaddhour, G., Andò, E., Salager, S., Bésuelle, P., Viggiani, C., Hall, S., & Desrues, J. (2013). Application of X-ray Tomography to the Characterisation of Grain-Scale Mechanisms in Sand. In Multiphysical Testing of Soils and Shales (pp. 195-200). Springer Berlin Heidelberg.

Kozicki, J., Donzé, F.V. (2009), Yade-open dem: an open-source software using a discrete element method to simulate granular material. Engineering Computations (26), pages 786–805.

Marzougui, D., Chareyre, B., Chauchat, J. (2015b), Numerical simulations of dense suspensions rheology using a DEM-Fluid coupled model, Geomechanics group annual report.

Marzougui, D., Chareyre, B., Chauchat, J. (2015a), Microscopic origin of shear stress in dense fluid-grain mixtures, Granular Matter (to appear).

Maurin, R., Chareyre, B., Chauchat, J., & Frey, P. (2013). Discrete element modelling of bedload transport. In *Two-pHase modElling for Sediment dynamIcs in Geophysical Flows* (pp. 6-p).

Papachristos, E., Scholtès, L., Donzé, F.-V., Chareyre, B. Pourpak, H. Hydraulic fracturation simulated by a 3D coupled HM-DEM model. (2015 - to appear) in *13th International ISRM Congress 2015*, 10 pages.

Richefeu, V., El Youssoufi, M. S., & Radjai, F. (2006). Shear strength properties of wet granular materials. *Physical Review E*, *73*(5), 051304.

Scholtès, L., Chareyre, B., Michallet, H., Catalano, E., Marzougui, D. (2015), Modeling wave-induced pore pressure and effective stress in a granular seabed. Continuum Mechanics and Thermodynamics (27), pages 305–323.

Scholtès, L., Chareyre, B., Nicot, F., Darve, F. (2009), Micromechanics of granular materials with capillary effects. International Journal of Engineering Science (47), pages 64–75.

Stránský, J. (2014). OPEN SOURCE DEM-FEM COUPLING. *in Booklet of the 1st YADE Workshop*, Grenoble, 2014.

Stránský, J., Jirásek, M. (2012). OPEN SOURCE FEM-DEM COUPLING. in Proc. 18th Int. Conf. Engineering Mechanics, Pragues, 2012, pp. 1237-1251.

Šmilauer, V. (2007), The splendors and miseries of yade design. in Annual Report of Discrete Element Group for Hazard Mitigation.

Šmilauer, V. (2014). DEM on GPU. *in Booklet of the 1st YADE Workshop*, Grenoble, 2014.

Šmilauer, V. (2010), Cohesive particle model using the discrete element method on the yade platform. PhD thesis at Czech Technical University in Prague, Faculty of Civil Engineering & Université Grenoble I – Joseph Fourier, École doctorale I-MEP2.

Šmilauer, V. Catalano, E. Chareyre, B. Dorofeenko, S. Duriez, J. Gladky, A. Kozicki, J. Modenese, C. Scholtès, L. Sibille, L. Stránský, J. Thoeni K. (2010), Yade Documentation. The Yade Project. (http://yade-dem.org/doc/)

Sweijen, T., Hassanizadeh, M., and Chareyre B. (2014), Pore-scale modeling of swelling porous media; application to superabsorbent polymers. In XX. International Conference on Computational Methods in Water Resources, Stuttgart, page 227.

Tejada, I.G., Sibille, L., Chareyre, B. (2015), Micromechanical modeling of internal erosion by suffusion in soils, Geomechanics group annual report.

Thoeni, K. (2013). *Performance test*, online content on the Yade-DEM website. https://yade-dem.org/wiki/Performance_Test

Toraldo, M., Chareyre, B., Sibille, L. (2015), Numerical modelling of the localized fluidization in a saturated granular medium using the coupled method DEM-PFV, Geomechanics group annual report.

Yuan C., Chareyre, B., Darve, F. (2015), Pore-scale simulations of two-phase flow in granular materials, Geomechanics group annual report .

Yuan, C., Chareyre, B., and Darve F. (2014), Pore-scale simulations of drainage for two-phase flow in dense sphere packings. *In XX. International Conference on Computational Methods in Water Resources*, Stuttgart, page 232.