European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2006
P. Wesseling, E. Oñate and J. Périaux (Eds)

A COMPARISON BETWEEN A DIRECT AND A MULTIGRID SPARSE LINEAR SOLVERS FOR HIGHLY HETEROGENEOUS FLUX COMPUTATIONS

A. Beaudoin[†], J.-R. de Dreuzy[††] and J. Erhel[*]

[*]INRIA/IRISA,Campus de Beaulieu, 35042 Rennes Cedex, France
e-mail: Jocelyne.Erhel@irisa.fr;web page:http://www.irisa.fr/sage/jocelyne
[†]LMPG,University of le Havre,25 rue Philippe Lebon,BP 1123,76063 Le Havre Cedex,France
e-mail: beaudoin@univ-lehavre.fr
[††]University of Rennes 1, UMR CNRS 6118, Campus de Beaulieu, 35042 Rennes Cedex, France
e-mail: Jean-Raynald.de-Dreuzy@univ-rennes1.fr

Key words: hydrogeology, heterogeneity, flux computation, sparse large linear systems, Cholesky factorization, multigrid, condition number.

Abstract. *Subsurface hydraulic properties are mainly governed by the heterogeneity of the porous medium considered. Our work aims at characterizing the asymptotic dispersion coefficients for highly heterogeneous permeability fields triggered by advection and constant local dispersion-diffusion. We have developed a fully parallel software for simulating flow and transport. We have compared two well-known sparse linear solvers, based respectively on a multifrontal Cholesky factorization and an iterative structured multigrid method. We study the effect of various parameters, mainly the system size, the number of processors and the degree of heterogeneity of the permeability field.*

1 INTRODUCTION

Solute transport in underground media is a key for both energy and environment applications like the storage of high-level nuclear wastes and the management of groundwater resources. Solutes are transported by advection and diffusion processes in highly heterogeneous media. Heterogeneity stems both from the juxtaposition of different lithologies (sand, clay, silt,) and from the differences within lithological units of sedimentary conditions [8, 14]. As a consequence physical properties like permeability are widely distributed. Depending on the geological settings, the distribution of permeability may be restricted to less than an order of magnitude [2] or span several orders of magnitude [22]. For low levels of heterogeneity, upscaling from the local scale heterogeneity typically at the decimetre scale to the macroscopic transport law typically at the kilometre scale has been solved analytically by use of linearization and limited-orders techniques [9]. For high levels of heterogeneity, such approximations cannot account for the high flow channelling and the

large range of flow values [Moreno and Tsang, 1994]. The macro-scale transport law relies in those cases on numerical modelling. The simulated domains must be larger than the critical spatial scale L_c at which the macroscopic asymptotic regime establishes. Numerical estimates of the asymptotic transport coefficients have not yet been determined without linearizing assumptions because domains of size larger than L_c have not been numerically attained [26]. In fact L_c is very large because the characteristic time scales of the involved phenomena (transport and diffusion) differ from at least 4 orders of magnitude [5]. Simulating simultaneously advection as the main transport phenomenon and diffusion as the only mixing process requires 2D domains of typical maximal scale over resolution scale of the order of 10^4 leading to around 10^8 elements.

In this work, we have developed software for computing the full velocity field and for simulating transport by advection-dispersion. Our platform is object-oriented, with a modular approach and clearly defined interfaces and calls to numerical free software libraries whenever possible. We run stochastic simulations based on a Monte-Carlo approach, with random samples of the permeability field. Our software is fully parallel and allows running large-scale simulations until the asymptotic behaviour. The flux computation is a linear model leading to a large sparse structured matrix. We have compared two well-known solvers, the direct solver PSPASES (multifrontal Cholesky factorization) and the iterative structured multigrid solver SMG in the HYPRE library. We study the effect of various parameters, mainly the system size, the number of processors and the variance of the permeability field.

The paper is organized as follows. We first define the physical and numerical model, then we describe the software; the fourth section is devoted to numerical experiments.

2 HYDRAULIC AND NUMERICAL MODEL

2.1 Physical model

The computational domain is a 2D rectangle with dimensions L_x and L_y. Permeability in porous media is classically modelled by a finitely-correlated field with lognormally distributed values [4, 9]. The permeability field is characterized by its mean m_y and covariance function C_y, given by

$$C_y(r) = \sigma^2 \exp\left(-\frac{|r|^2}{\lambda_y}\right), \tag{1}$$

with r the separation distance between two points, λ_y the correlation length and σ^2 the log-normal variance. The length λ_y is typically in the range $[0.1m, 100m]$ [28, 32] and the variance σ^2 is in the interval $[0, 7]$ that encompasses most of the generally studied values [19].

Classical laws governing the steady flow in a porous medium are mass conservation and Darcy law

$$v = -K\nabla h, \nabla . v = 0, \tag{2}$$

where K is the permeability field, v is the Darcy velocity and h is the hydraulic head. Boundary conditions are homogeneous Neumann on upper and lower sides and Dirichlet $h = 0$ on left side, Dirichlet $h = 1$ on right side. Therefore L_x and L_y are respectively longitudinal and transversal to the mean flow direction. The system should not be too much elongated in order to avoid border effects. In fact the velocity field close to the lateral boundaries is highly influenced by the no-flow boundary condition [24]. Transport equations are governed by advection and diffusion; the velocity field is thus an input for solving these equations. Simulations are divided into three main steps: we first generate the domain and the permeability field; then we solve the steady flow equations and compute the velocity field; finally, we solve the transient transport equations.

2.2 Numerical model

The flow equations are discretized on a regular grid using a classical finite-volume scheme, equivalent for a regular grid to a mixed finite-element scheme [3, 12]. The mesh size Δx is fixed as 1/8 to 1/10 of the correlation length which turns out to be enough for flow and transport studies [1]. Thus the number of elements is given by $N = L_x L_y/(\Delta x)^2$. Discrete linear equations are a linear system $Ax = b$, with a sparse structured penta-diagonal matrix A of order N, where x is the discrete hydraulic head and b comes from Dirichlet boundary conditions. Transport is simulated by a particle tracker whose key advantages for this study are the absence of numerical diffusion and the good performances for obtaining a not too precise solution [29]. Particles are transported along the flow lines and perform random jumps according to diffusion. In this Lagrangian framework, the transport equation is discretized with an explicit scheme [20]. The time step is adapted to the typical velocities on the boundaries of the mesh so that the particles perform on average ten steps in the mesh. Within the mesh, particle velocities are obtained by bilinear interpolation of the boundary velocities, because it is the sole interpolation method that respects mass conservation [21]. Establishing the asymptotic transport coefficients does not require a very precise solution but the simulation of flow and transport on a large number of permeability fields. The precision of the results depends both on the particle number and on the number of simulations.

3 PARALLEL OBJECT-ORIENTED SOFTWARE

We have developed an object-oriented software which provides a generic platform to run Monte-Carlo numerical simulations of flow and transport in highly heterogeneous porous media. The software is segmented in four main modules, respectively dedicated to the generation of the permeability field, the computation of flow, the computation of transport and the management of multiple simulations and their results. This modularity allows a great flexibility and portability. For example, it is easy to test different sparse solvers. All data are distributed on a cluster of processors and all computations are done in parallel. The unique source code is written in C++ and can be implemented on machines

with Unix, Linux or Windows systems. Graphical functions are written with OpenGL and parallel programming relies on the MPI library. The software integrates open-source components and will be made available to the community through an open-source licence.

3.1 Data distribution

Because large arrays cannot be stored on a single processor, they are distributed from the beginning to the end of the simulation, according to a domain decomposition. Three decompositions are implemented, respectively in slices along L_x or L_y or by square blocks of constant size. Each processor owns a well-defined part or the array corresponding to a subdomain and keeps in local memory one layer of cells surrounding its subdomain. These ghost cells allow to reduce communication costs between neighbouring processors and enables a fast computation of flow. The necessary methods have been gathered in a C++ class. Values stored on a processor are encapsulated within the class and can be accessed neither from outside nor from another processor. In fact no function requires more than the values stored in a cell and in its neighbours. The main public functions are the generation and the construction of the array, the extraction of a value and the statistical computations.

3.2 Permeability generation

The generation of the correlated lognormal field is performed via a Fourier transform with the software FFTW [**?**]. This library has a variety of composable solvers representing different FFT algorithms and implementation strategies, whose combination into a particular plan for a given size can be determined at runtime according to the characteristics of the machine/compiler in use. This peculiar software architecture allows FFTW to adapt itself to almost any machine and to have good performances. The construction of the array ends up with filling up the ghost cells, requiring the management of some communication between the processors. Permeability, velocity components and head values are all stored on the same types of array.

3.3 Flow computation

The second part of the software performs the flow computation taking as input the permeability array and delivering as output the head and velocity fields. The main functions are the derivation of the linear system from the permeability field, the use of the chosen solver giving the head field and the computation of the velocity field from the head field.

The discrete flow equations are a linear system $Ax = b$, where A is a symmetric positive definite sparse structured matrix. To study scale effects, mostly for heterogeneous media with a high variance, system dimensions should reach $L_x = 16384\Delta x$ and $L_y = 8192\Delta x$, resulting to a system size $N = 13410^6$. Parallel computing is essential to achieve these sizes, in order to satisfy both memory and CPU requirements. The condition number is related to the heterogeneities considered and increases very rapidly with the variance.

Several methods and solvers exist for these linear systems. They can be divided into three classes: direct, iterative or semi-iterative [16, 23]. Direct methods are highly efficient but require a large memory space. Iterative methods of Krylov type require less memory but need a scalable preconditioner to remain competitive. Iterative methods of multigrid type are often efficient and scalable, well-suited to regular grids, used by themselves or as preconditioners, but are sensitive to condition numbers [6, 31]. Semi-iterative methods such as subdomain methods are hybrid direct/iterative methods which can be good tradeoffs [27, 30]. For iterative and semi-iterative methods, the convergence and the accuracy of the results depend on the condition number which can blow up at large scale for a high variance. Thus there is no clear method of choice and the most suitable solver will depend on several parameters such as the system size, the variance, the computing architecture. In this paper, we compare a direct method and an iterative multigrid method; both methods are very efficient but have different drawbacks, either memory requirements increasing with domain size or CPU requirements increasing with heterogeneity. We use numerical libraries which are free, heavily used, portable, parallel. Because the matrix is positive definite, we choose PSPASES as a direct solver [13, 11]; we choose HYPRE and more precisely SMG (Structured MultiGrid) as a multigrid solver [7]. We use them as black boxes, with parameters set by default. In particular, we do not use any scaling or tuning.

3.4 Transport computation

The third part is the particle tracker that implements the transport computation. The particle tracker is separated from the remaining of the software by a pure virtual class that acts as an interface ensuring that the particle tracker can be used with other types of grids. The interface contains the minimal number of functions necessary from the array. The most important functions are the extraction of the velocities on the limit of the grid, the geometry of the mesh, the topology of the grid (i.e. which cells neighbour a given cell) and the presence of a boundary close to a cell.

3.5 Simulation supervision

This module controls the execution of Monte-Carlo simulations: it creates the parameters, calls the three modules described above, gathers the simulation results and computes the statistical ouputs.

4 NUMERICAL EXPERIMENTS AND RESULTS

4.1 Numerical tests

Our software enables us to run simulations at very large scales, with very high variances. We have defined two main physical parameters: the domain size $L_x = L_y$ (we consider mainly squares) and the variance σ^2. For each value, we generate several random samples in order to produce statistical results. We have obtained asymptotic behaviour and can now analyze the macro dispersion. Two examples of simulation are illustrated on Figure

1, for two values of the variance ($\sigma^2 = 0.5$ and $\sigma^2 = 3$) and the same size ($L_x = 512$), with no molecular diffusion. Although the permeability fields (top left) have the same geometric distribution, the colour scales are different, with a much larger interval for $\sigma^2 = 3$; the heterogeneity has as strong impact on the velocity field (longitudinal bottom left, transversal bottom right) and on the cloud of particles, which are clearly distinct (top right). We use 10 000 particles in our simulations, leading to results which do not vary more than 1% for 100 simulations. In the next subsections, we present performance results for a domain size ranging from $L_x = 256$ to $L_x = 4096$ and for a variance ranging from $\sigma^2 = 0.5$ to $\sigma^2 = 6$. All tests are performed on a SUN cluster composed of two nodes of 32 computers each. Each computer is a 2.2 Ghz AMD Opteron bi-processor with 2 Go of RAM. Inside each node, computers are interconnected by a Gigabit Ethernet Network Interface, and the two nodes are interconnected by a Gigabit Ethernet switch (CISCO 3750). This cluster is a component of the Grid'5000 computing resource installed at INRIA in Rennes, see Figure 2. For our simulations, we have used up to 128 processors.

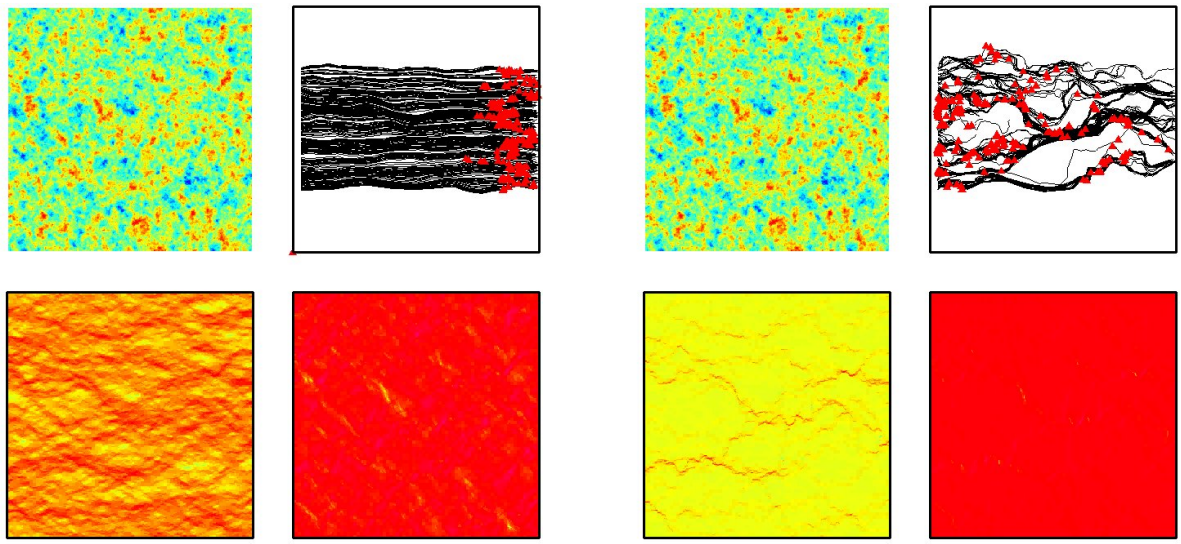

Figure 1: Examples of simulation with $\sigma^2 = 0.5$ (left) and $\sigma^2 = 3$. Top left is the permeability field, bottom left is the longitudinal velocity, bottom right is the transversal velocity, top right is the cloud of particles.

4.2 Complexity analysis

To analyze the complexity, we fix the variance $\sigma = 1$ and the number of processors $P = 2$ and vary the domain size, thus the matrix order N. Figure 3 represents timings and memory requirements with PSPASES. As predicted by the theory [13, 11], we observe on Figure 3 left, that the number of nonzeros in the Cholesky factor L is roughly proportionoal to $N \log N$ (we could also conclude to N), but much larger than the number of nonzeros in the original matrix A, which is $5N$. Clearly , the most CPU-intensive task is linear

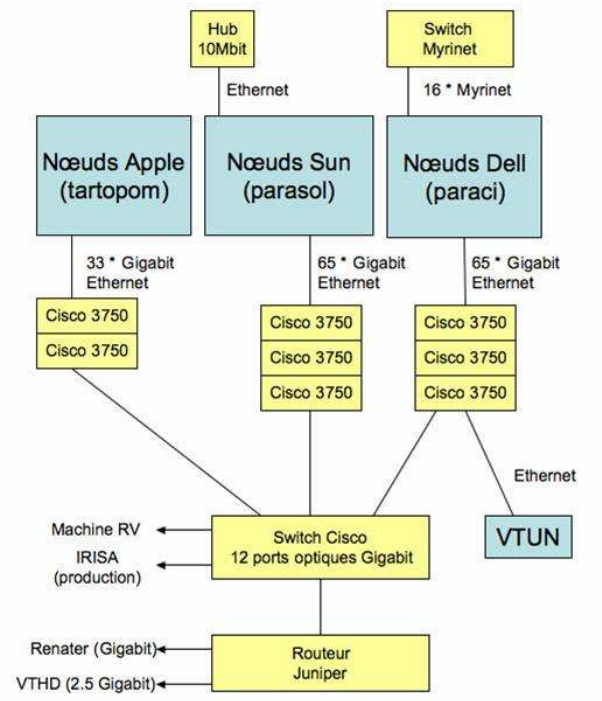

Figure 2: Architecture of the cluster and the grid used for simulations.

solving. Other tasks, such as matrix generation, have a linear complexity, while the direct sparse linear solver has a complexity in $O(N^{1.5})$. This is observed on Figure 3 right and confirms the theory for regular grids [13, 11]. Figure 4, left, represents the number of V-cycles in the multigrid method HYPRE/SMG, which is a measure of the convergence rate. The residual threshold is fixed to 10^{-13} and the number of V-cycles is limited to 1000. The number of V-cycles slightly increases with the domain size (the behaviour is roughly logarithmic), so that the CPU time (Figure 4, right) increases not only because of the system size but also because of the slower convergence. However, the complexity is here roughly linear in $O(N)$. We can conclude that, with a bi-processor computer, a multigrid method is more efficient than a direct method for very large matrices and small variances. Moreover, memory requirements are much lower, with a linear complexity in $O(N)$.

4.3 Scalability analysis

Now, we study the performances of both solvers on parallel computers. We keep the variance fixed $\sigma = 1$ and the domain size fixed (ranging from $N = 512$ to $N = 2048$) and we vary the number of processors P. CPU timings are represented on Figure 5 (left, small and medium size, right, large and very large size). For a small size, PSPASES is more efficient than HYPRE, for any number of processors. For a moderate size, both solvers give similar timings for a small number of processors, then PSPASES becomes faster. For

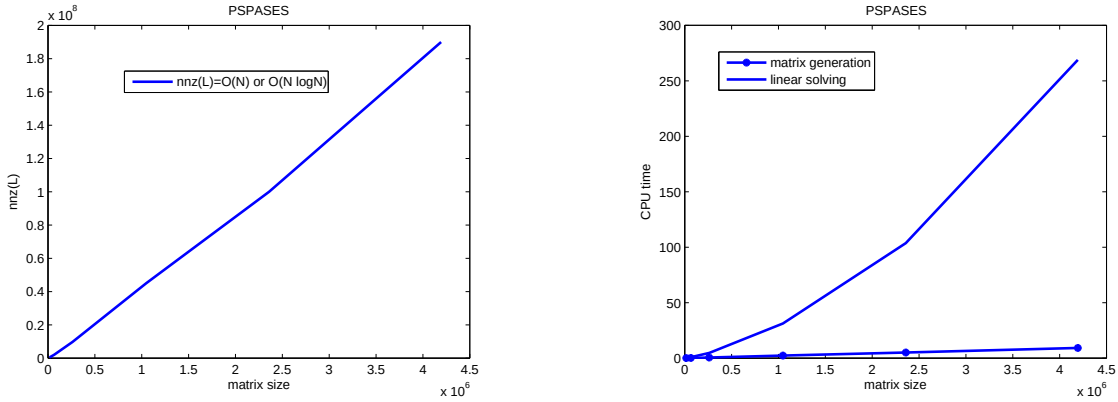

Figure 3: Complexity of PSPASES; memory requirements (left) and CPU time (right). Variable matrix order N, variance $\sigma = 1$, number of processors $P = 2$.

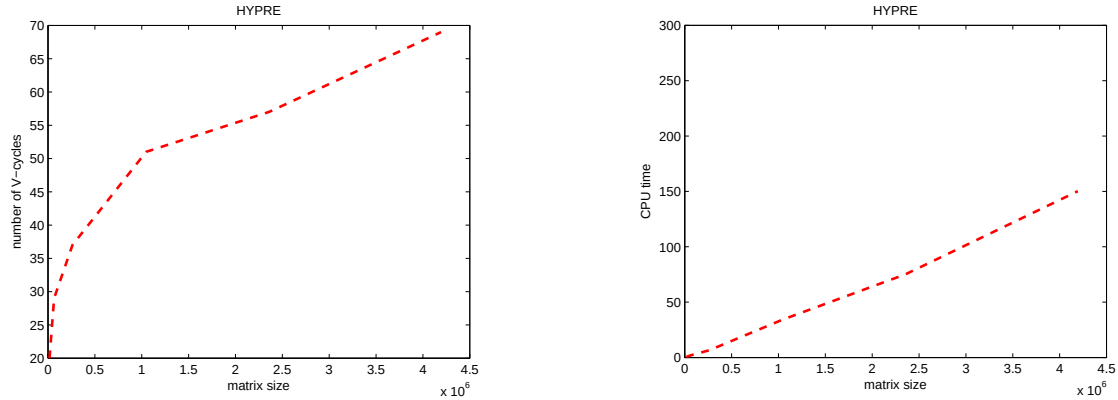

Figure 4: Complexity of HYPRE/SMG; convergence (left) and CPU time (right). Variable matrix order N, variance $\sigma = 1$, number of processors $P = 2$.

a large size, HYPRE is faster with a few processors and slower with many processors (the threshold is here at 32 processors and depends on the size and the architecture). For a very large size, PSPASES can run only on many processors because of memory requirements and is much slower than HYPRE.

Speed-ups are reported in Figure 6; since PSPASES must run on at least 2 processors, the speed-up is taken as $2T(2)/T(P)$, where $T(P)$ is the time with P processors. Also, the speed-up with HYPRE and the very large size is taken as $4T(4)/T(P)$. The speed-up increases with the size, because both solvers are in some sense scalable. However, PSPASES has a better parallel efficiency than HYPRE. This explains why CPU curves intersect on Figure 5. We have estimated the efficiency of PSPASES by computing the ratio $R = (N^{1.5}/(PT(P)))$, which is approximately proportional to the efficiency $E = T(1)/PT(P)$. In Table 1, we observe that this ratio, thus the efficiency, is roughly constant when N/P is kept constant. This is in good agreement with the theoretical result about

the isoefficiency of PSPASES [13, 11], showing that this solver is scalable in the sense that the efficiency is constant for a constant ratio N/P [15]. As far as HYPRE/SMG is concerned, our results do not allow to conclude to some isoefficiency or scalable speed-up.

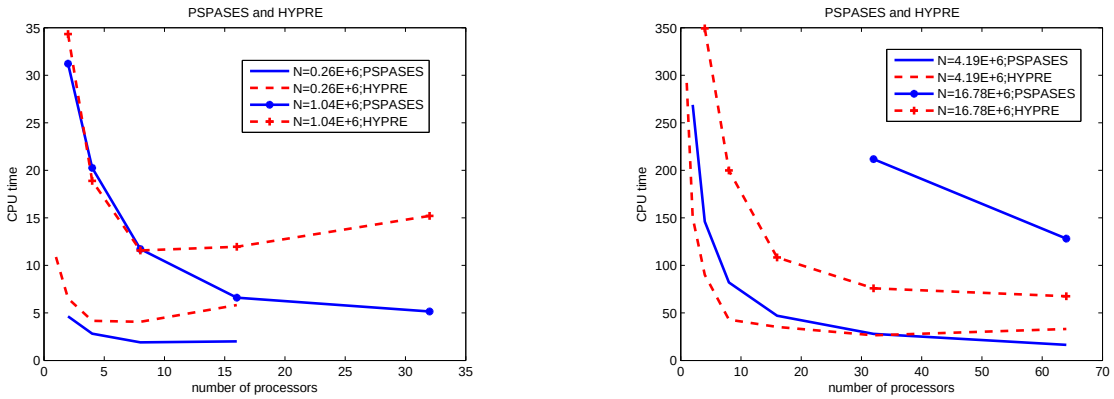

Figure 5: Parallel CPU time of PSPASES (blue solid lines) and HYPRE (red dotted lines). Variable number of processors P, variance $\sigma = 1$, small matrix order N (left), large matrix order N (right).

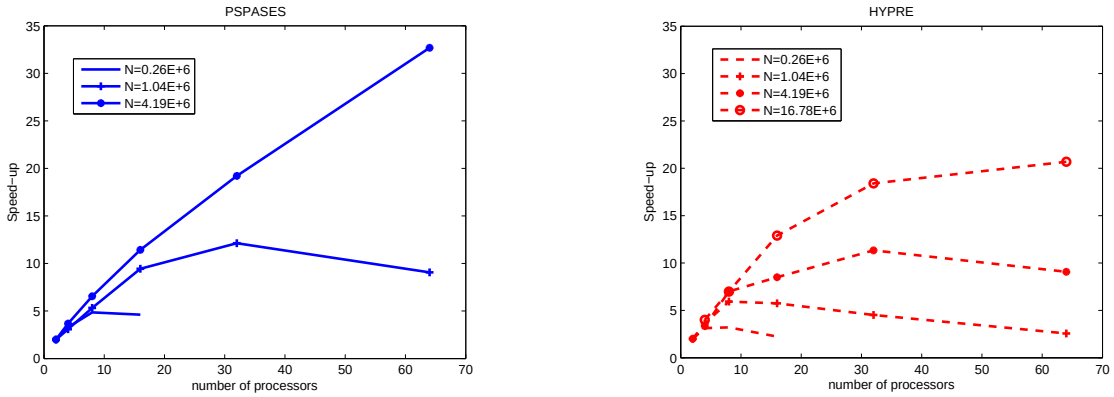

Figure 6: Speed-up of PSPASES (blue solid lines) and HYPRE (red dotted lines). Variable number of processors P, variance $\sigma = 1$, small matrix order N (left), large matrix order N (right).

4.4 Impact of heterogeneity analysis

Up to now, we simulated moderately heterogeneous media; now, we study highly heterogeneous permeability fields, with a variance ranging from $\sigma^2 = 0.5$ to $\sigma^2 = 6$. We keep the matrix order N and the number of processors P fixed. In Figure 7, left, we plot the number of V-cycles in the multigrid HYPRE/SMG solver. It increases very rapidly, from a few dozen up to more than 1000, which was the limit value in our settings. Accordingly, the CPU time (Figure 7, right, red dotted line) increases very rapidly. On the other hand,

P	N	T_p	R
2	262144	5.60	11977373
8	1048576	11.33	11844656
32	4194304	25.70	10443374
4	262144	2.92	11502234
16	1048576	6.06	11079774
64	4194304	13.08	10535895

Table 1: Values of the parameter R for various values of (P, N) and for PSPASES.

the CPU time of the direct PSPASES solver (Figure 7, right, blue solid line) remains constant, showing that the variance has almost no impact on the performance. We observe a small increase of the residual (one order of magnitude). However, the condition number increases with the variance, so we can expect a loss of accuracy in the solution. The first conclusion drawn from this experiment is that the multigrid method (as used in our simulations) is not efficient at all for a high variance. The direct solver remains efficient, but still requires heavy computations and memory space with very large domain sizes. There are several ways to improve accuracy and convergence of the multigrid method. A first step will be to scale the matrix in order to reduce the condition number [25]. A second step will be to use other multigrid methods, with a smoother tuned for heterogeneous data. A third step will be to use domain decomposition methods, with interface conditions also adapted to a highly heterogeneous permeability field or Krylov iterative methods with deflation [10, 18, 17].

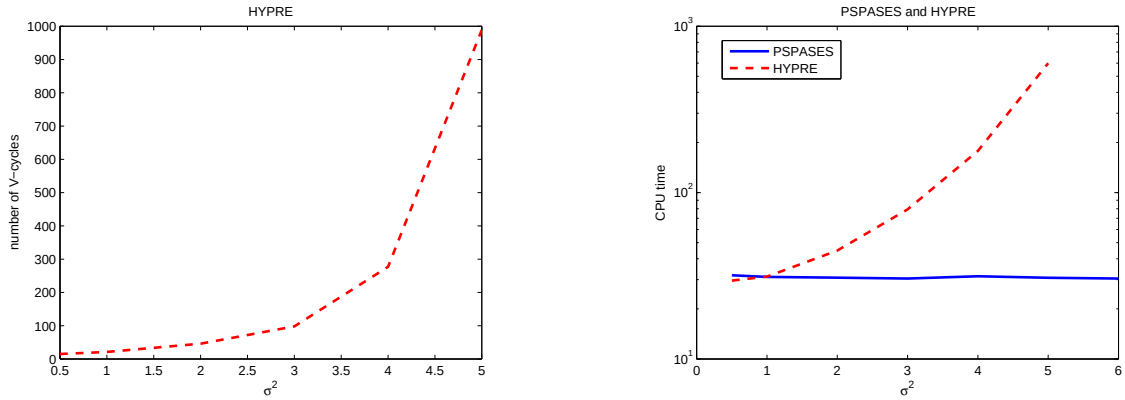

Figure 7: Convergence of HYPRE/SMG (left) and CPU time (right) of PSPASES (blue solid line) and HYPRE/SMG (red dotted line). Variable variance σ^2, matrix order $N = 1024 * 1024 = 1.0410^6$ (left), number of processors $P = 4$.

5 CONCLUSION

Thanks to our parallel software, we can simulate flow and transport in 2D highly heterogeneous porous media with very large size. In our simulations, the direct solver PSPASES appear to be very efficient for small to moderate sizes whereas the iterative multigrid solver HYPRE/SMG is faster for very large sizes. But PSPASES gives better parallel performances, so that the threshold between both methods depends on the number of processors. Also, SMG is very sensitive to the degree of heterogeneity, which increases the condition number, and becomes less efficient than PSPASES for highly heterogeneous media. We plan to investigate in more details this sensitivity and to test different methods, such as diagonal scaling, other smoothers, domain decomposition. We are currently gathering the statistical outputs in order to analyze the macro-dispersion effects. Also we plan to develop a 3D version and to run samples of Monte-Carlo simulations on several clusters interconnected in a computational grid.

REFERENCES

[1] R. Ababou, D. Mclaughlin, L. W. Gelhar, and A. F. B. Tompson. Numerical simulation of three-dimensional saturated flow in randomly heterogeneous porous media. *Transport in Porous Media*, 1989.

[2] J.M. Boggs, S.C. Young, Beard, L.M., L.W. Gelhar, K.R. Rehfeldt, and E.E. Adams. Field study of dispersion in heterogeneous aquifer. 1. overview and site description. *Water Resources Research*, 28(12):3281–3291, 1992. Tanguy.

[3] G. Chavent and J.E. Roberts. A unified physical presentation of mixed, mixed-hybrid finite elements and standard finite difference approximations for the determination of velocities in waterflow problems. *Advances in Water Resources*, 14(6):329–348, 1991.

[4] G. Dagan. *Flow and Transport in Porous Formations*. Springer Verlag, 1989.

[5] M. Dentz, H. Kinzelbach, S. Attinger, and W. Kinzelbach. Temporal behavior of a solute cloud in a heterogeneous porous medium, 1, point-like injection. *Water Resources Research*, 36(12), 2000.

[6] C. C. Douglas and M. B. Douglas. MGNet Bibliography. Department of Computer Science and the Center for Computational Sciences, University of Kentucky, Lexington, KY, USA and Department of Computer Science, Yale University, New Haven, CT, USA, 1991–2002 (last modified on September 28, 2002); see http://www.mgnet.org/mgnet-bib.html.

[7] R. D. Falgout, J. E. Jones, and U. Meier Yang. Pursuing scalability for hypre's conceptual interfaces. *ACM Trans. Math. Softw.*, 31(3):326–350, 2005.

[8] Alan R. Freeze and John A. Cherry. *Groundwater*. Prentice Hall, 1979.

[9] Lynn W. Gelhar. *Stochastic Subsurface Hydrology*. Prentice Hall. Engelwood Cliffs, New Jersey, 1993.

[10] I. G. Graham and P. O. Lechner. Domain decomposition for heterogeneous media. In *domain decomposition methods DD16*, 2005.

[11] A. Gupta, F. Gustavson, M. Joshi, G. Karypis, and V. Kumar. Pspases: An efficient and scalable parallel sparse direct solver. In Tianruo Yang, editor, *Kluwer International Series in Engineering and Computer Science*, volume 515, 1999.

[12] Willem Hundsdorfer and Jan Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2003.

[13] M. Joshi, G. Karypis, V. Kumar, A. Gupta, and F. Gustavson. Pspases : Scalable parallel direct solver library for sparse symmetric positive definite linear systems : User's manual. Technical Report 97-059, Department of Computer Science, University of Minnesota, Minneapolis, MN., 1997.

[14] C. E. Koltermann and S. M. Gorelick. Heterogeneity in sedimentary deposits: A review of structure-imitating, process-imitating, and descriptive approaches. *Water Resources Research*, 32(9), 1996.

[15] V.P. Kumar and A. Gupta. Analyzing scalability of parallel algorithms and architectures. *Journal of Parallel and Distributed Computing*, 22(3):379–391, 1994.

[16] G. Meurant. *Computer solution of large linear systems*. North Holland, Amsterdam, 1999.

[17] R. Nabben and C. Vuik. A comparison of Deflation and Coarse Grid Correction applied to porous media flow. *SIAM J. Numer. Anal.*, 42:1631–1647, 2004.

[18] R. Nabben and C. Vuik. A comparison of deflation and the balancing preconditioner. *SIAM J. Sci. Comput.*, 27:1742–1759, 2006.

[19] Shlomo P. Neuman and Shlomo Orr. Prediction of steady state flow in nonuniform geologic media by conditional moments: Exact nonlocal formalism, effective conductivities and weak approximation. *Water Resources Research*, 29(2):341–364, 1993.

[20] Dubravka Pokrajac and Ranko Lazic. An efficient algorithm for high accuracy particle tracking in finite elements. *Advances in Water Resources*, 25(4):353–369, 2002. TY - JOUR.

[21] D. W. Pollock. Semianalytical computation of path lines for finite-difference models. *Ground Water*, 26(6):743–750, 1988.

[22] K. R. Rehfeldt, J. M. Boggs, and L. W. Gelhar. Field study of dispersion in a heterogeneous aquifer, 3, geostatistical analysis of hydraulic conductivity. *Water Resources Research*, 28(12), 1992.

[23] Y. Saad. *Iterative methods for sparse linear systems*. PWS Publishing Company, 1996.

[24] P. Salandin and V. Fiorotto. Solute transport in highly heterogeneous aquifers. *Water Resources Research*, 34(5):949–961, 1998.

[25] M. H. Schneider and S. Zenios. A comparative study of algorithms for matrix balancing. *Operations Research*, 38:439–455, 1990.

[26] H. Schwarze, U. Jaekel, and H. Vereecken. Estimation of macrodispersion by different approximation methods for flow and transport in randomly heterogeneous media. *Transport in Porous Media*, 43(2):265–287, 2001.

[27] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[28] Andrew F. B. Tompson, Robert D. Falgout, Steven G. Smith, William J. Bosl, and Steven F. Ashby. Analysis of subsurface contaminant migration and remediation using high performance computing. *Advances in Water Resources*, 22(3):203–221, 1998. TY - JOUR.

[29] Andrew F.B. Tompson and Lynn W. Gelhar. Numerical simulation of solute transport in three-dimensional, randomly heterogeneous porous media. *Water Resources Research*, 26(10):2541–2562, 1990.

[30] A. Toseli and O. Widlund. *Domain decomposition methods-algorithms and theory*. springer series in computational mathematics, 2005.

[31] P. Wesseling. *An Introduction to Multigrid Methods*. Edwards, 2004.

[32] A.D Woodbury and E.A. Sudicky. The geostatistical characteristics of the borden aquifer. *Water Resources Research*, 27(533-546), 1991.