Computational Geosciences
https://doi.org/10.1007/s10596-022-10136-8

ORIGINAL PAPER

GPU implementation of Explicit and Implicit Eulerian methods with TVD schemes for solving 2D solute transport in heterogeneous flows

Lucas Bessone[1,2,3] · Pablo Gamazo[1] · Marco Dentz[4] · Mario Storti[5] · Julián Ramos[1]

Received: 3 October 2020 / Accepted: 25 January 2022

Abstract

In this work we present an efficient implementation of Eulerian TVD methods. We apply parallelization strategies based entirely on GPU for the solution of the 2D transport equation in heterogeneous porous media. Additionally, a parallel strategy is proposed for the generation of exponentially correlated lognormally distributed permeability fields in GPU. The programs are developed using C++/CUDA. The implemented methods are used to solve advective dominant problems, in a context of Monte Carlo type simulations to numerically determine the longitudinal and transversal macrodispersion coefficients averaging over 100 simulations for permeability fields for a large range of variances. The following types of transport are considered for testing: pure advection, advection-diffusion and advection-dispersion. The performance in terms of the computation time of explicit and implicit methods are compared. We show that the implemented algorithms allow to efficiently solve problems in computational domains of up to 134.5 million cells in a single GPU.

Keywords GPU · TVD · Eulerian methods · Transport equation · High Performance Computing

1 Introduction

Flow and transport in natural and engineeporous media are complex due to multi-scale spatial heterogeneity in the hydraulic medium properties. Thus, accurate and efficient numerical methods are pivotal to reproduce, understand and predict these processes in a broad array of applications ranging from the assessment of groundwater contamination, soil remediation [16], the design and safety assessment of underground storage of radioactive waste [38], as well as geothermal energy production and geological carbon-dioxide storage [36].

In this paper, we focus on the development of a GPU-based simulation tool for single phase flow and conservative transport in continuum scale heterogeneous porous media. The flow is described by the divergence-free Darcy equation [4]

$$\mathbf{u} = -\mathbf{K}\nabla H, \quad \nabla \cdot \mathbf{u} = 0, \tag{1a}$$

where $\mathbf{u}$ is the Darcy velocity, $\mathbf{K}$ the hydraulic conductivity tensor, and H is hydraulic head. The singular complexity of flow and transport in porous media resides in the fact that hydraulic conductivity may vary over 12 orders of magnitude in different porous material ranging from granite

✉ Pablo Gamazo
gamazo@unorte.edu.uy

Lucas Bessone
lcbessone@gmail.com

Marco Dentz
marco.dentz@csic.es

Mario Storti
mario.storti@gmail.com

Julián Ramos
jramos@unorte.edu.uy

[1] Departamento del Agua, CENUR LN, Universidad de la República (UDELAR), Salto, Uruguay

[2] Departamento de Matemática y Estadística del Litoral, CENUR LN, UDELAR, Salto, Uruguay

[3] Universidad Tecnológica Nacional - Facultad Regional Concordia, Concordia, Argentina

[4] Spanish National Research Council (IDAEA-CSIC), Barcelona, Spain

[5] Centro de Investigación de Métodos Computacionales (CIMEC), CONICET-UNL, Santa Fe, Argentina

Published online: 07 March 2022

to gravel [4]. Advective-dispersive transport of a passive scalar C is described by the partial differential equation [4]

$$\frac{\partial C}{\partial t} + \nabla \cdot (\mathbf{u}C - \mathbf{D}\nabla C) = 0, \tag{1b}$$

where $C(x, t)$ is the concentration of the solute [kg/m^3], $\mathbf{u}(x) = (u_x, u_y)$ is the velocity field [m/s] and $\mathbf{D}$ is the hydrodynamic dispersion tensor [m^2/s] . There are different numerical methods for the simulation of flow and non-reactive transport in groundwater. The optimal strategy for numerically solving the system of Eq. 1a are in general problem-dependent. For advection-dominated problems, Lagrangian methods [37] have advantages over Eulerian methods because they are in general free of numerical diffusion, and non-physical oscillations [51]. Lagrangian methods are ideal for purely advective problems. However, in the presence of diffusion the use of a great number of particles might be necessary in order to accurately reproduce the concentration field, which can lead to a high computational load. This represents a challenge for the simulation of non-linear reactive transport problems using Lagrangian methods, which require the simultaneous representation of the concentration fields of multiple chemical species. There are a series of approaches that address these issues, including streamline based methods of characteristics [13, 21], which combine particle tracking and finite differences, the implementation of efficient kernel-density estimators [19] and the development of efficient numerical particle tracking implementations [41].

We center on purely numerical methods to solve the system of Eq. 1a. Combined with a suitable scheme for the reduction of numerical diffusion, Eulerian methods are well suited for the solution of coupled and non-linear flow and transport problems in heterogeneous media. In particular, they allow for the straightforward implementation of algorithms to solve problems that involve the interaction between mobile and immobile phases, and the coupling between flow and transport (variable density, variable viscosity, variations in porosity, etc.). Eulerian methods can solve transport in the same mesh as flow, and they do not require the transformation of particles mass or numbers to concentrations.

To deal with numerical diffusion in advection-dominated problems, high-resolution (HR) schemes have emerged in a total variation diminishing (TVD) context [25, 26]. These conservative schemes allow simulating the transport of pulses and/or injection lines, reducing significantly the numerical diffusion as well as spurious oscillations due to numerical diffusion errors. These techniques are computationally more expensive than standard Eulerian methods and, like those, require the use of very fine grids to obtain a good representation in the simulation of this type of problems [8]. The relatively recent development of the general purpose graphics processing units (GPUs), and the computational power they offer, have made these methods an attractive alternative for solving advection dominated transport problems in heterogeneous media. The use of GPUs for scientific computation has emerged as a viable option for high-performance computing since it allows simulating, with good resolution and in relatively short times, problems involving millions of cells. In order to obtain an efficient GPU code, algorithms must be designed according to the specific characteristics of the hardware [10, 53, 56]. The execution model adopted by the GPU for parallel computation is that of Single Instruction Multiple Threads (SIMT) [33]. This involves dividing the processing of a mesh into different functions (CUDA kernels), which group together a set of instructions that are passed to the GPU so that each processor thread executes the same instruction but on a large data set [50]. Regarding temporal discretization, in general implicit methods are preferred due to their greater stability throughout the evolution of the problem because they are able to use much longer time steps than explicit schemes. However, the characteristics of the SIMT model in GPUs makes using explicit methods computationally attractive, event though it requires considering a much smaller time step [34].

In this paper, we develop and implement an efficient GPU based solver for flow and transport in heterogeneous porous media. We apply this solver for the systematic study of macrodispersion in heterogeneous media using a stochastic modeling approach. This requires the numerical solution of a suite of flow and transport simulations for different scenarios. Most works in the literature have studied macrodispersion using Lagrangian methods [5–7, 12, 14, 15, 17, 18, 22, 24, 27, 30, 31, 44–46]. Some authors have used hybrid methods such as Trefry et al. [52], who use the Eulerian-Lagrangian method presented in [43]. Ramasomanana et al. [39] use the hybrid ELLAM - Eulerian-Lagrangian localized adjoint method [57]. To the best of our knowledge, there is no record in the literature of macrodispersion studies carried out considering purely Eulerian methods. This may be due to the fact that these methods can be computationally demanding, since they require a large number of cells to decrease numerical diffusion and high resolution schemes for the advection term. In this work, we show that macrodispersion in highly heterogeneous media can be efficiently quantified on modern GPU processors using Eulerian methods. We compare the efficiency, in terms of the calculation time required for the simulation, of an explicit algorithm and an implicit algorithm, since the latter have disadvantages over the former given the GPU SIMT execution model.

The work is organized as follows. In Section 2 the governing equations and numerical methods are reviewed. In Section 3 the details of the implementation of the algorithms in GPU are given. In Section 4 we apply the solvers for the study of macrodispersion in heterogeneous porous media. Finally, in Section 5 we present the main conclusions and contributions of this work.

2 Governing equation and numerical method

In this section we present the governing equations for flow, its characterization and the non-reactive solute transport in porous media.

2.1 Flow equation

The steady state equation for incompressible fluids mass conservation has the following expression: $\nabla\cdot \mathbf{u}(x) = 0$. The flow equation in a porous medium is obtained by combining the Darcy law and divergence-free condition of Eq. 1a

$$\nabla\cdot [\mathbf{K}(x)\nabla H(x)] = 0 \tag{2}$$

The most used methods to transform (2) into a system of algebraic equations are Finite Elements (FEM), Finite Differences (FD), and Finite Volumes (FVM). In this work, the equation is solved for two-dimensional domains using cell-centered FVM, considering uniform Cartesian grids (which is equivalent to FDM). Without loss of generality, a rectangular domain Ω and a regular mesh discretization ($\Delta x = \Delta y = h$) are considered. In the following, the cell center and the respective neighboring cells are designated by subscript C and E, W, N, S respectively. The lowercase subscripts indicate cell faces centers (see Fig. 1). Thus the sum over $f \sim nb(C)$ indicates that expression is evaluated in centers of face f on all the faces that belong to cell C, that is, faces neighboring the center C. Similarly $F \sim NB(C)$ indicates that the expression is evaluated at all the centers of cell F neighboring cell C.

If we integrate member to member (2) in cell C, and we apply the divergence and mean value theorems, we obtain:

$$\int\limits_{V_C} \nabla\cdot [\mathbf{K}(x)\nabla H(x)]\, dV =$$

$$\int\limits_{\partial V_C} \mathbf{K}(x)\nabla H(x)\cdot \mathbf{dS} = \sum_{f\sim nb(C)} (\mathbf{K}(x)\nabla H(x))_f \cdot \mathbf{S}_f = 0 \tag{3}$$

where $\mathbf{S}_f$ is the normal vector outgoing to the face f. Since hydraulic conductivity is variable in the domain, a proper way to evaluate $\mathbf{K}(x)\nabla H(x)$ in the center of a face is using the harmonic average between the values of the cells that share the face (C and F):

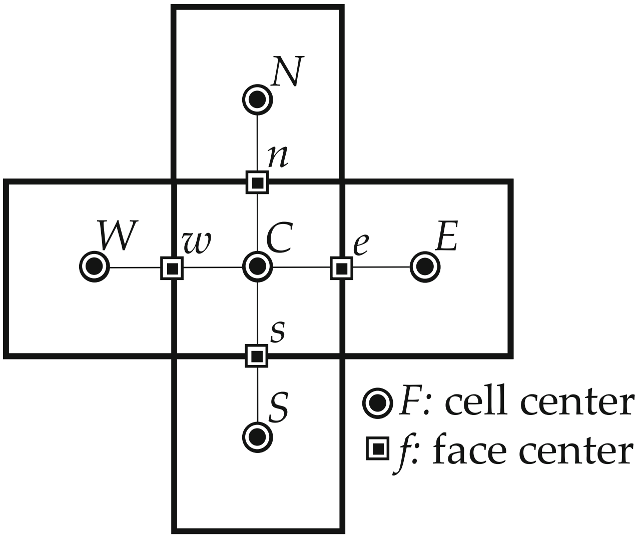


Fig. 1 Computational molecule

$$\mathbf{K}_f = \left(\frac{1}{\frac{1}{2}\left(\frac{1}{K_C}+\frac{1}{K_F}\right)}\right) = \left(\frac{2K_C K_F}{K_C+K_F}\right) \tag{4}$$

finally, using a centered spatial scheme to approximate the gradient, (3) can be rewritten discretely for cell C as:

$$\sum_{F\sim NB(C)} \left(\frac{2K_C K_F}{K_C+K_F}\right)\left(\frac{H_F-H_C}{h}\right) h = 0 \tag{5}$$

By considering all the cells of the domain, a linear system of algebraic equations is obtained. In this case the matrix of the resulting system is symmetrical and positively defined, therefore a suitable method to solve it is the Conjugate Gradients (CG) method. Since the system matrix is very ill conditioned, the use of preconditioners is required to achieve convergence at acceptable computation times. In this work we implement the Preconditioned Conjugated Gradients (PCG) solver in GPU using two types of suitable preconditioners to parallelize: diagonal scaling (Jacobi's preconditioner) and a Truncated Neumann Series based preconditioner [23]. Designated respectively as D and TN, in each case the preconditioning matrix looks like:

$$M_D = D, \quad M_{TN1} = \left(I + LD^{-1}\right) D \left(I + (LD^{-1})^T\right) \tag{6}$$

where I is the identity matrix, D is the diagonal part and L the strict lower triangular part of the system matrix A. Since A is symmetric, both preconditioners are symmetric. The inversion of the system in the first case is trivial, while for the second preconditioner, the truncated approximation of the Neumann series is applied. In effect, the factor $\left(I+LD^{-1}\right)^{-1}$ can be defined with the series:

$$I-LD^{-1}+\left(LD^{-1}\right)^{2}-\left(LD^{-1}\right)^{3}+\dots$$

now we choose the truncated series of first order as the inverse of the preconditioner:

$$M_{TN1}^{-1}=\left(I-D^{-1}L^{T}\right)D^{-1}\left(I-LD^{-1}\right) \tag{7}$$

Defining $E = I - LD^{-1}$, one can write $M_{TN1}^{-1} = E^{T}D^{-1}E$. Then for the implementation, the matrix D is explicitly stored and then the inversion of the system $Mz = r$ (i.e. $z = M^{-1}r$) is carried out in two operations: first the product $z_{\text{tmp}} = D^{-1}Er$ and then the product $z = E^{T}z_{\text{tmp}}$. By using M_{TN1} the computing time are reduced by a factor of 0.6 with respect to the use of Jacobi's preconditioner for the cases studied in this work.

2.2 Transport equation

Transport is modeled using the conservation law Eq. 1b. The dispersion tensor is defined by [4]

$$\mathbf{D}=\frac{1}{\sqrt{u_x^2+u_y^2}}\begin{bmatrix}\alpha_L u_x^2+\alpha_T u_y^2 & (\alpha_L-\alpha_T)\,u_x u_y\\ (\alpha_L-\alpha_T)\,u_x u_y & \alpha_L u_y^2+\alpha_T u_x^2\end{bmatrix}+\begin{bmatrix}D_m & 0\\ 0 & D_m\end{bmatrix} \tag{8}$$

where α_L and α_T are the longitudinal and transverse dispersivities respectively in [m] ; and D_m is the molecular diffusion coefficient in [m^2/s].

As before, by integrating and using the divergence and mean value theorems, (1b) can be rewritten as:

$$\left(\frac{\partial C}{\partial t}\right)_C V_C=\sum_{f\sim nb(C)}\left(\mathbf{D}_f\nabla C_f\cdot\mathbf{S}_f-\dot{m}_f C_f\right) \tag{9}$$

where $V_C = h^2$ is the volume of the cell for the two-dimensional case and $\dot{m}_f = \mathbf{u}_f\cdot\mathbf{S}_f$ is the mass flow that crosses the face of the cell. Spatial discretization arises by approximating the gradient operator and the values of the unknown in faces C_f. For the first, it is enough to use the centered differences scheme $(\nabla C)_f = (C_F - C_C)/h$, while for the second term a better estimate is necessary to obtain numerical stability and precision in the results, especially in dominant advective problems. A first option could be to use the *upwind* scheme, which in the case of a non-uniform velocity field is expressed as:

$$\dot{m}_f C_f = ||\dot{m}_f, 0||C_C - || - \dot{m}_f, 0||C_F \tag{10}$$

where $||a, b||$ is used to indicate $\max(a, b)$. This scheme considers a linear combination of the values in cell centers according to the flow direction, and is stable in dominant adevective problems. However, this is a first-order approximation and it smoothes out the solution excessively by generating a numerical diffusion equal to $|\mathbf{u}_f|h/2$. On the other hand, the well-know Godunov's theorem states that: *Linear numerical schemes for solving partial differential equations, having the property of not generating new extreme (monotone scheme), can be at most first-order accurate*. That is, if a second order o greater scheme is needed, for it to preserve monotony if must be non-linear. Therefore the solution at C_f depends on the value at C_C and its neighbors C_F (with $F \sim NB(C)$) non-linearly as we will show later. In the next subsection we briefly review high resolution (HR) schemas to approximate the advective term in Eq. 9.

2.2.1 Total Variation Diminishing (TVD) framework

High-order upwind biased interpolations fill the gaps in precision of the upwind schema. But such approaches, according to the Godunov [20] theorem, provide generally unbounded solutions, which can be particularly problematic in non-uniform velocity fields or highly heterogeneous flows. A bounded high-order scheme known as a High Resolution (HR) scheme, is obtained by imposing the TVD condition that we explain below.

Considering for simplicity a one-dimensional mesh, the Total Variation (TV) is defined as $TV(C) = \sum_i |C_{i+1} - C_i|$ where i is the index of a node of mesh. A scheme is then said to be TVD if TV does not grow over time:

$$TV(C^{t+\Delta t}) \le TV(C^{t}) \tag{11}$$

A monotonous high-order scheme is TVD and a TVD scheme in turn preserves monoticity [25]. Limiting functions (limiter or flux limiter) can be used to build a TVD scheme. Such limiters prevent the occurrence of non-physical oscillations. Using the approach of Sweby [49], calling the limiter $\psi(r)$ where r indicates the ratio between two consecutive gradients, the value in the center of a face can be obtained in a simple way:

$$C_f = C_C + \frac{1}{2}\psi(r_f)\left(C_D - C_C\right)\ ;\ \text{with } r_f = \frac{C_C - C_U}{C_D - C_C} \tag{12}$$

where the subscripts D and U indicate the upwind and downwind nodes, and therefore must be chosen according to the direction that the velocity u_f has in the center of the face. For example, the advective flow corresponding to the face *east* results in:

$$\begin{aligned}\dot{m}_e C_e &= \left[C_C + \frac{1}{2}\psi(r_e^+)\,(C_E - C_C)\right] ||\dot{m}_e, 0|| \\ &\quad - \left[C_E + \frac{1}{2}\psi(r_e^-)\,(C_C - C_E)\right] ||-\dot{m}_e, 0||; \\ \text{with } r_e^+ &= \frac{C_C - C_W}{C_E - C_C},\; r_e^- = \frac{C_E - C_{EE}}{C_C - C_E} \end{aligned} \tag{13}$$

Many TVD schemes have been developed in this way, by appropriately choosing the limiter $\psi(r_f)$ (OSHER [9], MUSCL [54], SUPERBEE and MINMOD [42] among others). Note that whenever $\psi(r)$ allows a TVD schema to be generated, C_f will be a non-linear combination of the values in cell centers, since r_f is a function of C_C and neighbors C_F. In other cases the scheme will be linear, but not TVD, as is the case with HO schemes, for example: *central scheme* ($\psi(r) = 1$), *second order upwind* scheme ($\psi(r) = r_f$) or even QUICK [32] scheme ($\psi(r) = (r_f + 3)/4$). This creates a complication for implicit methods since it would require an iterative strategy to solve the nonlinear system that is generated.

2.2.2 Deferred Correction (DC) approach

In this work, two time schemes are considered, one explicit and the other implicit. For the explicit method the limiting function for the linearization of the advective scheme $\psi(r_f)$ can be calculated using the values of the previous temporary solution C^t. For the implicit algorithm, this implies the off-diagonal coefficients in the system to have opposite signs, thus violating a basic rule for the stability of iterative algorithms [35]. Although the above can be circumvented if a direct method is used, such a situation is not at all practical in discretizations with millions of cells.

To deal with the aforementioned problem, in the implicit algorithm we use the deferred correction technique [29]. The strategy is to treat the advective term as follows:

$$\dot{m}_f C_f^{HR} = \underbrace{\dot{m}_f C_f^U}_{\text{implicit}} + \underbrace{\dot{m}_f \left(C_f^{HR} - C_f^U\right)}_{\text{explicit}} \tag{14}$$

where the superscripts U and HR indicate the use of the upwind and TVD schemas respectively (ie, use Eq. 10 for U and Eq. 13 for HR). The idea then is that the value in the center of the face is treated implicitly considering the first order scheme. The difference between the upwind and TVD schemes are treated explicitly, calculating them based on the last available solution, that is, the one obtained on the previous iteration within an iterative process. The advantage of this approach is that a matrix with diagonal dominance is obtained in the implicit temporal schema (desirable in iterative solvers, such as Krylov's methods), while the non-linearity of the TVD schema is dealt explicitly.

2.2.3 Spatial and temporal discretization

The spatial discretization of Eq. 9 consists of using the DC and HR schemas for the first and second terms, respectively. The implementation was based on the MINMOD [42] schema. For the temporal discretization the theta (θ) method was applied, which gives rise to:

$$\begin{aligned} &\left(\frac{C^{t+\Delta t} - C^t}{\Delta t}\right)_C h^2 = \\ &(1-\theta)\left[h \sum_{F\sim NB(C)} \mathbf{D}_f \left(\frac{C_F - C_C}{h}\right) - \sum_{f\sim nb(C)} \dot{m}_f C_f^{HR}\right]^t \\ &+\theta \left[h \sum_{F\sim NB(C)} \mathbf{D}_f \left(\frac{C_F - C_C}{h}\right) - \sum_{f\sim nb(C)} \dot{m}_f C_f^{HR}\right]^{t+\Delta t} \end{aligned} \tag{15}$$

where $\theta \in [0, 1]$, so that for $\theta = 0$ we obtain the explicit scheme of order $O(\Delta t)$, known as forward differences or Forward Euler (FE). For $\theta = 1/2$ we obtain the implicit scheme known as centered differences or Crank Nicolson (CN) with precision of $O(\Delta t^2)$. To simplify the notation, superscripts will be used so that C_C^n corresponds to the variable evaluated in the center of the cell C in current time $t = n\Delta t$.

2.2.4 Explicit scheme

Replacing $\theta = 0$ in Eq. 15, and grouping the terms we obtain the explicit method or FE:

$$\begin{aligned} C_C^{n+1} &= C_C^n \\ &+\frac{\Delta t}{h^2}\left[\sum_{F\sim NB(C)} \mathbf{D}_f (C_F^n - C_C^n) - \sum_{f\sim nb(C)} \dot{m}_f C_f^{n,HR}\right]; \\ \text{with } \dot{m}_f C_f^{n,HR} &= \left[C_C^n + \frac{1}{2}\psi(r_f^{n,+})\left(C_F^n - C_C^n\right)\right] ||\dot{m}_f, 0|| \\ &\quad - \left[C_F^n + \frac{1}{2}\psi(r_f^{n,-})\left(C_C^n - C_F^n\right)\right] ||-\dot{m}_f, 0|| \end{aligned} \tag{16}$$

where $\psi(r_f^{n,+})$ y $\psi(r_f^{n,-})$ are the limiting function evaluated by the ratios as in Eq. 13, according to the direction of the velocity and using the values of the previous time $t = n\Delta t$. Note that the resulting expression is nonlinear in the values of C but can be solved completely explicitly. The analytical CFL stability constraint for this algorithm, for the case of isotropic dispersion (diagonal $\mathbf{D}$), is:

$$\Delta t^{TVD}_{\max} \leq \frac{h^2}{\sum\limits_{f\sim nb(C)}\left[\mathbf{D}_f + ||\dot{m}_f, 0||(1-\psi(r_f^+)/2) - ||-\dot{m}_f, 0||\psi(r_f^-)/2\right]} \tag{17}$$

where $\mathbf{D}_f$ is the corresponding component of the diagonal of the dispersion tensor evaluated in center of face. As can be seen, the time step is a function of $(\nabla C)_f$ when the TVD scheme is considered. On the other hand, since the velocity field in space is not uniform, the following restriction must be considered:

$$\Delta t \leq \min_{i,j}\left\{\Delta t^{(i,j)}_{\max}\right\}\ ,\ 0 \leq i \leq N_X,\ 0 \leq j \leq N_Y \tag{18}$$

2.2.5 Implicit scheme

For the CN scheme, $\theta = 1/2$ is replaced in Eq. 15, resulting in a nonlinear equation that is solved iteratively by applying the deferred correction technique (see Eq. 14). In this way, at each time step a non-linear system must be solved through the following iterative scheme:

$$\begin{aligned}&\frac{h^2}{\Delta t}C_C^{(k+1)} - \frac{1}{2}\sum_{F\sim NB(C)}\mathbf{D}_f\left(C_F^{(k+1)} - C_C^{(k+1)}\right)\\ &+\frac{1}{2}\sum_{f\sim nb(C)}\dot{m}_f C_f^{(k+1),U} = -\frac{1}{2}\sum_{f\sim nb(C)}\dot{m}_f\left(C_f^{(k),HR} - C_f^{(k),U}\right)\\ &\frac{1}{2}\left[\frac{h^2}{\Delta t}C_C^n + \sum_{F\sim NB(C)}\mathbf{D}_f\left(C_F^n - C_C^n\right) - \sum_{f\sim nb(C)}\dot{m}_f C_f^{n,HR}\right]\end{aligned} \tag{19}$$

We consider the superscript (k) for the internal iteration solutions, choosing the seed $C^{(0)} = C^n$ at each time step. After reaching convergence, we consider $C^{n+1} = C^{(k+1)}$. Note that a linear system of equations must be solved for each iteration and that the nonlinear component is taken into account in the last summation on the right side. Regarding the CFL restriction, in this case it has the expression:

$$\Delta t^{TVD}_{\max} \leq \frac{2h^2}{\sum\limits_{f\sim nb(C)}\left[\mathbf{D}_f + ||\dot{m}_f, 0||(1-\psi(r_f^+)/2) - ||-\dot{m}_f, 0||\psi(r_f^-)/2\right]} \tag{20}$$

2.3 Random conductivity field generation

The conductivity fields were generated in GPU using the method proposed by [47, 48]. The method considers that the random process can be represented as the sum of series of cosines with a random frequency. The lognormally distributed conductivity field is determined from: $\mathbf{K}(x) = \mathbf{I}\exp(f(x))$ were:

$$f(x) = \sqrt{\frac{2}{N}}\sigma^2_{\ln K}\sum_{i=1}^{N}\cos\left(k_1^{(i)}x + k_2^{(i)}y + \theta^{(i)}\right) \tag{21}$$

here $\sigma^2_{\ln K}$ is the variance of $\ln(K)$; (x, y) are the coordinates of a point in the computational domain; $k^{(i)}_{1,2}$ are sampled according to a joint probability density function chosen according to the type of correlation required. $\theta^{(i)} \sim U(0, 2\pi)$ is sampled using a uniform distribution. For the exponential correlation case, $k^{(i)}_{1,2}$ must be sampled according to the two-dimensional Cauchy-Lorentz distribution:

$$p(k_1, k_2) = \frac{1}{2\pi}\frac{\lambda^2}{\left[(k_1\lambda)^2 + (k_2\lambda)^2 + 1\right]^{3/2}} \tag{22}$$

To get k_1 and k_2 we look for a cumulative distribution function (CDF) of Eq. 22 integrating p in $\mathbb{R}^2$. Using the change of variables $k_1 = \rho\cos\phi/\lambda$ and $k_2 = \rho\sin\phi/\lambda$, with the following Jacobian for the transformation ρ/λ^2, results in:

$$\begin{aligned}&\iint\limits_{\mathbb{R}^2}\frac{1}{2\pi}\frac{\lambda^2\ dxdy}{\left[(k_1\lambda)^2 + (k_2\lambda)^2 + 1\right]^{3/2}} =\\ &\lim_{R\to\infty}\int_0^{2\pi}\int_0^{R}\frac{1}{2\pi}\frac{\rho\ d\phi d\rho}{(\rho^2+1)^{3/2}} = \lim_{R\to\infty}\left[1 - \frac{1}{\sqrt{R^2+1}}\right] = 1\end{aligned} \tag{23}$$

then the CDF is $\Theta(r) = 1 - \frac{1}{\sqrt{r^2+1}} = u$, and its inverse:

$$r = \left[\frac{1-(1-u)^2}{(1-u)^2}\right]^{1/2} \tag{24}$$

By choosing $u_i \sim U(0, 1)$ y $\phi_i \sim U(0, 2\pi)$ uniformly distributed, $k^{(i)}$ are terminated by:

$$k_1^{(i)} = r_i\cos\phi_i/\lambda,\ k_2^{(i)} = r_i\sin\phi_i/\lambda \tag{25}$$

The generated fields have the following correlation function:

$$C_{\exp}(r) = \sigma^2_{\ln K}\ exp\left(-\frac{|r|}{\lambda}\right) \tag{26}$$

where $|r|$ is the distance between two points. The degree of heterogeneity is governed by the statistical properties λ and $\sigma_{\ln K}$.

3 GPU implementation details

3.1 Hardware

The implemented algorithms were run on two computers, one with a previous-generation GPU (K40) and the other with a last-generation GPU (V100).

1. Computer 1: Dell PowerEdge R720 (2013/14): 2 Intel Xeon (R) CPU E5-2620v2 processors (6 cores each, 2.1GHz), 128GB DDR3 RAM. 2 NVIDIA Tesla K40 GPUs (2880 CUDA cores, running at 745MHz, 12GB GDDR5 memory), connected via PCI-e bus.
2. Computer 2: Dell PowerEdge R740 (2018/19): 2 Intel Xeon Gold CPU 6138 processors (20 cores each, 2.0GHz) with 128GB DDR4 RAM. 1 NVIDIA Tesla V100 GPU (5120 CUDA cores, running at 1230MHz, 32GB HBM2 memory), connected via PCI-e.

3.2 Software

The operating system of both computers is CentOS Linux v7, the GCC compiler version is 4.8.5. For the CUDA nvcc compiler, v9.1 was used on computer 1 and v10.1 was used on computer 2. All implementations were performed in DP (double precision). For linear algebra operations, the CUBLAS library [1] was used, and for the conductivity fields, the engines for the generation of random sequences of CURAND were used [2].

3.3 Parallelization of explicit method

The implementation of the explicit method was carried out considering that the data storage and all the calculations were performed on the GPU. On the other hand, to improve efficiency, the CUDA SIMT paradigm was considered and the mesh processing was divided into different CUDA kernels, that is, there is a kernel for each edge case and one for the interior cells (see Fig. 2) according to the following scheme:

- 4 kernels to process each vertex of the domain.
- 4 kernels to process edges.
- 1 kernel to process the interior of the domain.

in this way all these kernels can be executed concurrently since the results only depend on the data from the previous time step.

The strategy adopted for the implementation of the kernels is that each thread (CUDA threads) processes a point in the plane.Due to the low reuse of data in the operations involved we omitted the use of shared memory, and only registers and reads to global memory are used. Regarding the accesses to the data in global memory of the device, within the kernel 14 accesses are required for each cell (5 + 5 corresponding to C^{n+1} and C^n, 2 + 2 corresponding to u_x and u_y) in the case of pure advection and advection-diffusion. While in the case of advection-dispersion , 26 accesses are required (5+9 corresponding to C^{n+1} and C^n, 6 + 6 corresponding to u_x and u_y).The large number of accesses for the latter case is due to the need to interpolate the different components of the gradients on the faces:

$$\left(\frac{\partial C}{\partial y}\right)_e = \frac{1}{2}\left[\frac{(C_{NE}-C_{SE})}{2h} + \frac{(C_N - C_S)}{2h}\right] \tag{27}$$

It is also necessary to obtain all the components of the velocity field on the faces. For example, to obtain the component y of the velocity in east face, the values in faces n, s of cell C and those of the neighboring cell E, which we denote with subscripts nE and sE respectively:

$$\left(u_y\right)_e = \frac{1}{4}\left[\left(u_y\right)_n + \left(u_y\right)_s + \left(u_y\right)_{nE} + \left(u_y\right)_{sE}\right] \tag{28}$$

This reduces the performance of the solvers due to the required global memory accesses and will affect both algorithms (explicit and implicit) in practically the same way.

To illustrate the explicit method algorithm, the following notation will be used: C^*_{input}, C^*_{output}, u^*_x and u^*_y indicate pointers to vectors stored in the GPU's global memory. The algorithm 1 describes the explicit method.

Algorithm 1: Explicit algorithm for 2D transport equation.

```
initialization: $C^*_{\text{input}} \leftarrow C^n$; $C^*_{\text{output}} \leftarrow C^{n+1}$;
while $t < T_{final}$ do
    $t \leftarrow t + \Delta t$;
    compute_kernel_interior<<<grid,
      block>>>($C^*_{\text{output}}$, $C^*_{\text{input}}$, $u^*_x$, $u^*_y$);
    compute_kernel_edge_south<<<grid$_S$, block$_S$
      >>>($C^*_{\text{output}}$, $C^*_{\text{input}}$, $u^*_x$, $u^*_y$);
    compute_kernel_vertex_SW<<<1,
      1>>>($C^*_{\text{output}}$, $C^*_{\text{input}}$, $u^*_x$, $u^*_y$);
    ...
    sincronization: cudaDeviceSynchronize();
    compute moments for statistics from new
      solution: $C^{n+1}$;
    pointer interchange: $C^*_{\text{input}} \longleftrightarrow C^*_{\text{output}}$;
end
```

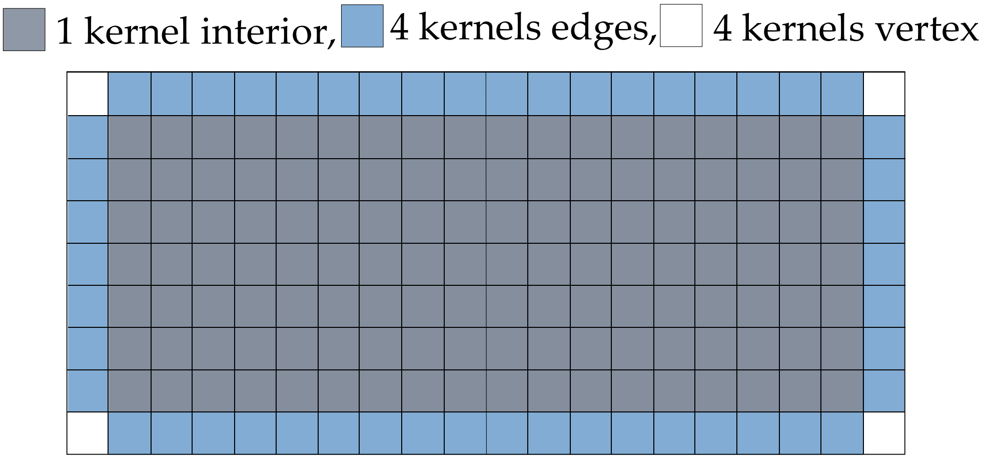


Fig. 2 Strategy for parallel processing via concurrent kernels

3.4 Parallelization of implicit method

For the implicit algorithm it is required to solve a linear system of equations $Ax = rhs$, and since the matrix is not symmetric, the BiCGStab [55] method was implemented in GPU using the *matrix free* style. In this way, the matrix of the system Ax required by the solver is not saved, that is, the operations required to compute the corresponding row of A are performed with the corresponding values of the vector x. The parallelization strategy of the linear operator Ax and the calculation of the right hand side is the same as in the explicit case, using procedures of the following type:

- kernel_linear_operator($C^{(k+1)*}_{\text{output}}$, $C^{(k+1)*}_{\text{input}}$, u^*_x, u^*_y) for the first,
- kernel_compute_RHS(rhs^*, C^{n*}, $C^{(k)*}$, u^*_x, u^*_y) for the second.

For the linear operator used in the BiCGStab solver, 14 and 26 accesses are required for each cell, as in the explicit case kernels depending on whether the dispersion is modeled or not. In the kernel for the computation of the right hand side, 19 accesses are required ($5+5+5$ corresponding to C^n, $C^{(k)}$ and rhs, 2+2 to read u_x and u_y) and 31 accesses ($9+5+5$ corresponding to C^n, $C^{(k)}$ and rhs, $6+6$ to read u_x and u_y) according to the transport case to simulate. The algorithm 2 describes the implementation of the implicit method.

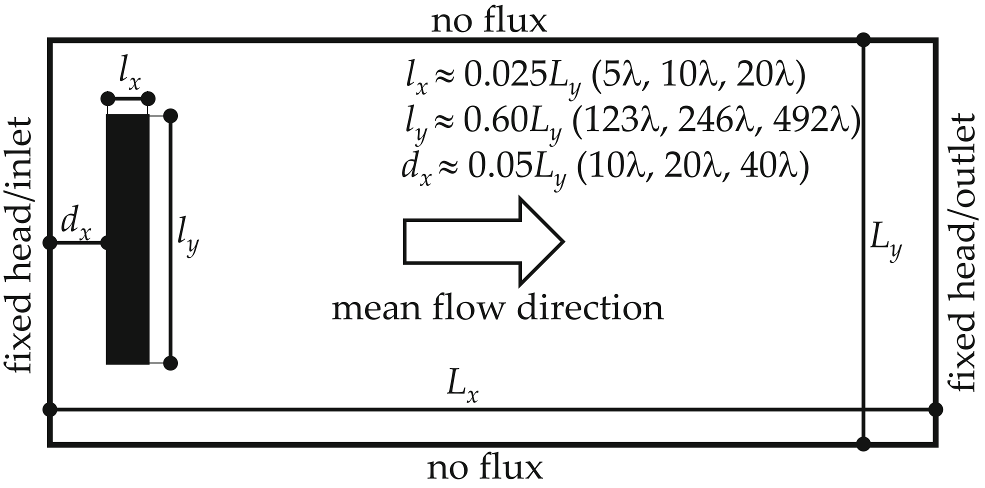
3.5 Parallelization of random field simulator

For the generation of the conductivity fields, two kernels were assembled, the first corresponds to the generation of the 3 vectors: $k^{(i)}_{1,2}$ and $\theta^{(i)}$, and the second for the computation of $f(x) = \ln \mathbf{K}$ according to (21). The calculation process is as follows: in the first kernel, each thread generates 3 random numbers with uniform distribution using the CURAND library. Then with the expressions (24) and (25) the 3 values $(k_{1,2}, \theta)$ can be determined. The number of threads for this kernel is equal to the N chosen in the equation (21). The strategy for

Table 1 Parameters and types of cases considered for the numerical simulations

Parameter/case	Values
$\sigma^2_{\ln K}$	0.25, 1, 2.25, 4, 6.25
Grid size $\Delta x = \Delta y = h$	5m
Resolution: $\lambda/\Delta x = \lambda/\Delta y$	10
Covariance model	Isotropic Exponential
Transport type	Pure advection $Pe = \infty$
	Advection diffusion $Pe_d = 100$
	Advection dispersion $Pe_L = Pe_T = 100$
$[L_x/\lambda, L_y/\lambda]$	[205,204.8] for $\sigma^2_{\ln K} \in \{0.25, 1\}$
	[410,410] for $\sigma^2_{\ln K} = 2.25$
	[820,820] for $\sigma^2_{\ln K} = 4$
	[1640,820] for $\sigma^2_{\ln K} = 6.25$
Inyection window	$l_x \times l_y = 0.025 L_y \times 0.60 L_y$
Numbers of simulations	N=100

Fig. 3 Domain setup and boundary condition (flow and transport equation respectively)


no flux
l_x
$l_x \approx 0.025L_y$ (5λ, 10λ, 20λ)
$l_y \approx 0.60L_y$ (123λ, 246λ, 492λ)
$d_x \approx 0.05L_y$ (10λ, 20λ, 40λ)
fixed head/inlet
d_x
l_y
mean flow direction
L_y
fixed head/outlet
L_x
no flux

Algorithm 2: Implicit algorithm for 2D transport equation with TVD using DC aproach.

```
initialization: $C^*_{\text{input}} \leftarrow C^n$;
while $t < T_{final}$ do
    $t \leftarrow t + \Delta t$;
    copy solution as seed: cudaMemcpy($C^{(k)*}$,
      $C^*_{\text{input}}$, cudaMemcpyDeviceToDevice);
    $dif \leftarrow 1.0$;
    $iter \leftarrow 0$;
    while $((dif > 1.0e-6)$ and $(iter < iter_{\max}))$
     do
        compute RHS: via concurrent kernels;
        sincronization: cudaDeviceSynchronize();
        call solver: BiCGStab($C^{(k+1)*}$, $rhs^*$);
        $dif \leftarrow ||C^{(k+1)} - C^{(k)}||_{L_\infty}$;
        $iter \leftarrow iter + 1$;
    end
    set new solution: $C^*_{\text{output}} \leftarrow C^{(k+1)*}$;
    compute moments for statistics from new
      solution: $C^{n+1}$;
    pointer interchange: $C^*_{\text{input}} \longleftrightarrow C^*_{\text{output}}$;
end
```

the second kernel is for each thread to read the three vectors previously generated and to apply the equation (21) according to the coordinates x and y corresponding to each node. For this kernel $4N$ GPU global memory read/write access are required, 3 for vectors $k_{1,2}$, θ and 1 writing for $f(x)$ for each iteration in a sum up to N, in a vector of size $N_x \times N_y$. Before finishing, the equivalent conductivity is calculated using the geometric average, for this the scalar product $d = \langle f(x), \text{ones}_{(N_x \times N_y)} \rangle = \sum(f(x))$ is calculated and then $K_{\text{geom}} = exp(d)$ is evaluated. Finally, the vector f is copied from GPU to CPU and at the moment of saving the fields in files, the operation $\mathbf{K} = exp(f(x))$ is carried out. This methodology is very efficient as well as being easy to implement. The algorithm 3 reflects the implementation used.

Algorithm 3: Random field simulator with exponential covariance.

```
random_number_gen_kernel<<<grid1,
  block1>>>($k_1^*$, $k_2^*$, $\theta^*$);
compute log_conductivity_kernel<<<grid,
  block>>>($k_1^*$, $k_2^*$, $\theta^*$, $f^*$);
compute dot prod in GPU:
  $K_{geom} \leftarrow exp(\langle f^*, \text{ones}\rangle)$;
copy $f^*$ to CPU from GPU: cudaMemcpy($f^*_{\text{host}}$, $f^*$,
  cudaMemcpyDeviceToHost);
save in file $K \leftarrow exp(f^*)$;
```

4 Case study: Evaluation of macro dispersion of a non reactive solution in highly heterogeneous media (validation and performance assessment)

In this section, we apply the methods presented above for the characterization of the macro dispersion in a set of cases with different degrees of heterogeneity. We validate the results obtained by the Eulerian-TVD method with results from literature (based on Lagrangian methods), and we compare the efficiency of the implicit and explicit schemes in terms of computation time. The numerical experiment consists of simulations of transport in 2-D domains for heterogeneous velocity fields calculated from log-normally distributed random hydraulic conductivity $K(x)$ fields with exponential covariance. From the simulations,

Table 2 Generation time for 2D random field on Equipment 2 (Nvidia Tesla V100 GPU 5120 cuda cores, Xeon Gold 6138 CPU - 40 cores)

Nx	Ny	N [Mcell]	avg. time [s]	Total time [min]
2050	2048	4.19	0.431	0.7
4100	4100	16.81	1.615	2.7
8200	8200	67.24	6.083	10.1
16400	8200	134.48	12.51	20.9

The time corresponds to the average for each realization and total time correspond to the generation of 100 cases

the statistics for the longitudinal and transverse macro dispersion coefficient were evaluated using the moment method.

4.1 Computational domain setup

The same computational grid was used for the generation of the conductivity field, the calculation of flow and the simulation of transport. The dimensions of the domain for each case were defined as a function of $\sigma_{\ln K}$, considering the appropriate size to correctly obtain the asymptotic values of the macro-dispersion coefficients [6, 15, 17, 39, 52]. We consider the following cases $L_x \times L_y \in \{205\lambda \times 204.8\lambda,\ 410\lambda \times 410\lambda,\ 820\lambda \times 820\lambda,\ 820\lambda \times 1640\lambda\}$ (were λ is the correlation length), resulting in grids from 4.2 to 134.4 million cells. The statistics were based on Monte Carlo (MC) type experiments considering 100 random conductivity fields for each case studied. To avoid border effects, the initial condition for transport is a high concentration window centered vertically in the dimensions domain $l_x \times l_y \approx 0.025 L_y \times 0.6 L_y$ and located at a distance $d_x \approx 0.05 L_y$ from the flow inlet (left edge of the domain). Table 1 summarizes the parameters used in this work.

For the flow equation, a Neumann no-flow boundary condition is considered for the north and south border, and Dirichlet boundary conditions are set in the east and west border. For the transport equation, zero flow is considered at the north and south border, and only advective flow is considered the east and west border (see Fig. 3).

4.2 Conductivity fields and mesh discretization

In this work the conductivity fields $\mathbf{K}(x)$ were generated for variances $\sigma_{\ln K} \in \{0.25, 1.00, 2.50, 4.00, 6.25\}$, considering a mesh discretization of $\Delta x = \Delta y = h$, where $\lambda = 10h$. For the generation of the fields $N = 10000$ was considered in (21). Table 2 shows the generation times for the GPU conductivity fields for different grid sizes. Reported times include the following tasks: random field generation and geometric mean computation (both on GPU), copy GPU to CPU and save in files for later reading. As can be seen, the times obtained are comparable in performance, to those obtained using other methods [40]. Figs. 4, 5 and 6 show

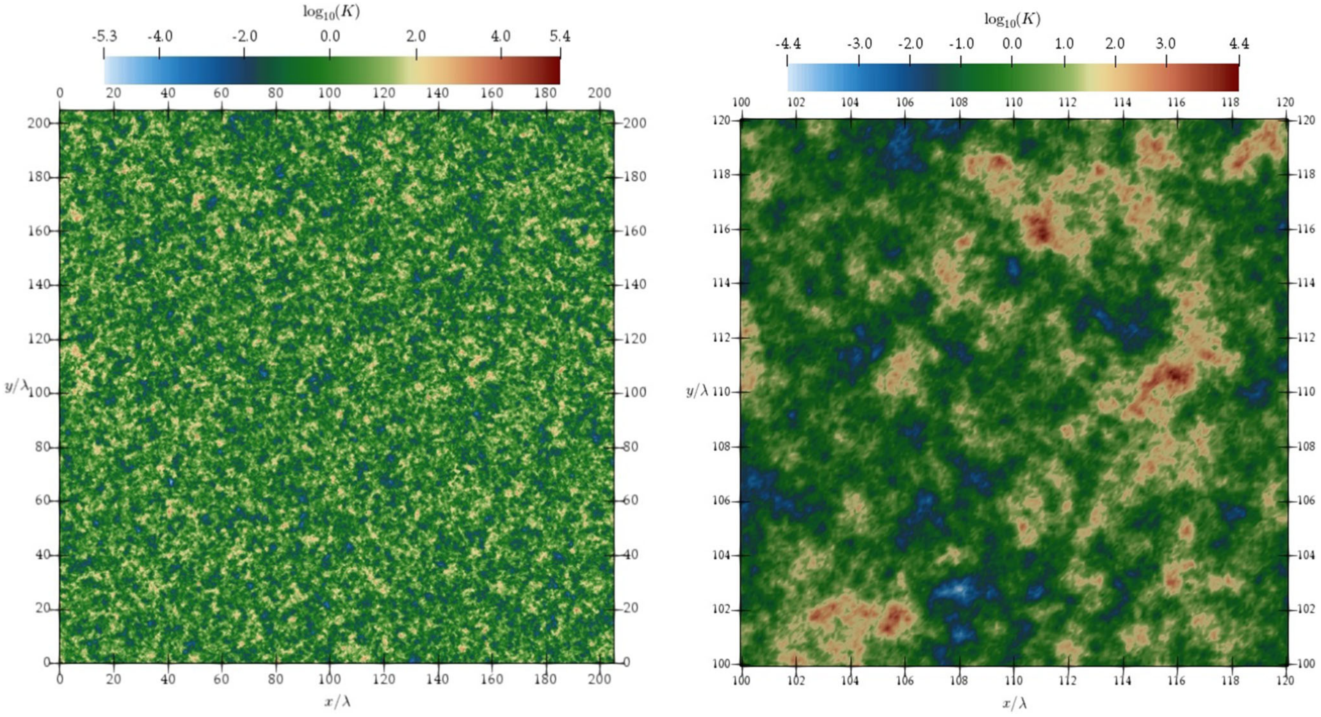


Fig. 4 Logonormal conductivity field with variance $\sigma^2_{\ln K} = 6.25$; correlation length $\lambda = 10$; exponential covariance; for entire domain and an inner window of $[20x/\lambda \times 20x/\lambda]$

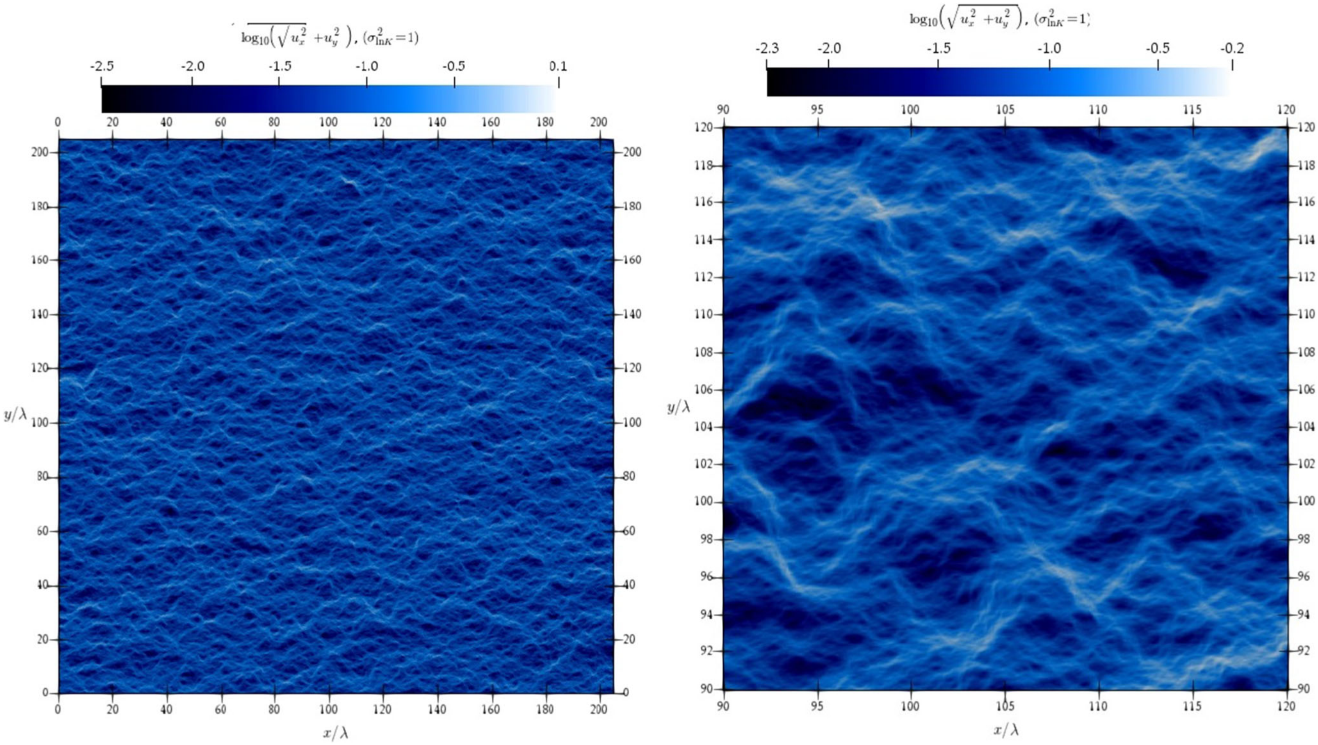


Fig. 5 Velocity field simulate for lognormal conductivity field with variance $\sigma^2_{\ln K} = 1$; correlation length $\lambda = 10$; exponential covariance; for entire domain (left) and an inner window of $[30x/\lambda \times 30x/\lambda]$ (right)

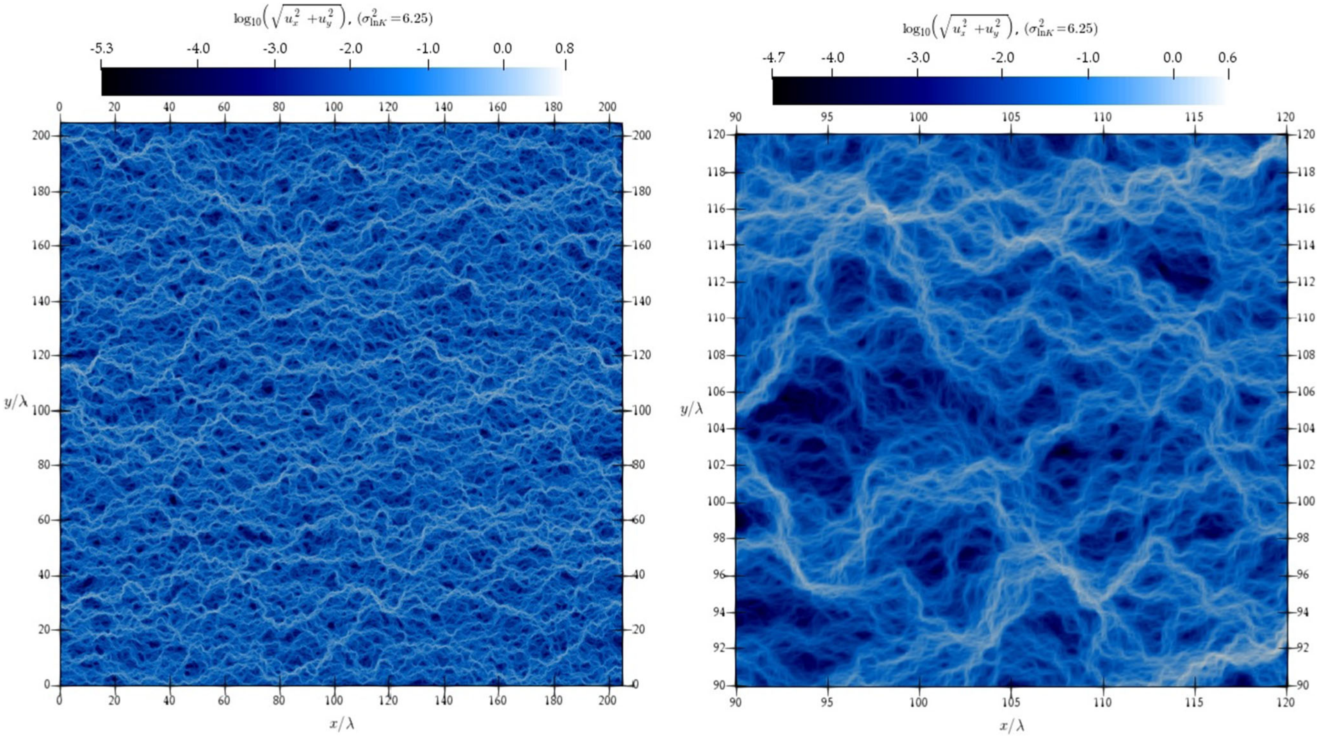


Fig. 6 Velocity field simulate for lognormal conductivity field with variance $\sigma^2_{\ln K} = 6.25$; correlation length $\lambda = 10$; exponential covariance; for entire domain (left) and an inner window of $[30x/\lambda \times 30x/\lambda]$ (right)

Fig. 7 Comparison of effective permeability defined as being the mean flux (v_m) normalized by the total head gradient versus geometric mean of conductivity (K_g) for 100 different samples. Exponential (top) and gaussian (bottom) covariance models are shown. Domain of [410λ × 410λ], resolution $\lambda/\Delta x = \lambda/\Delta y = 10$

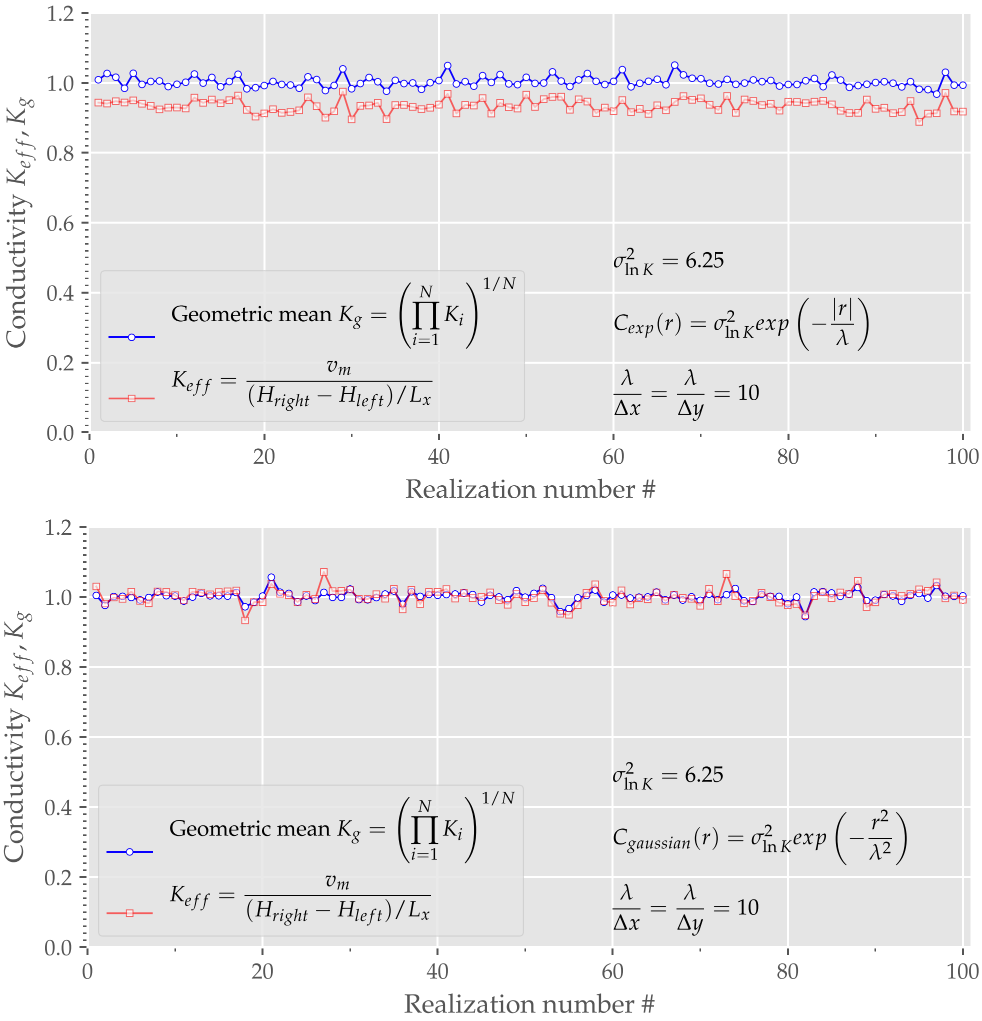
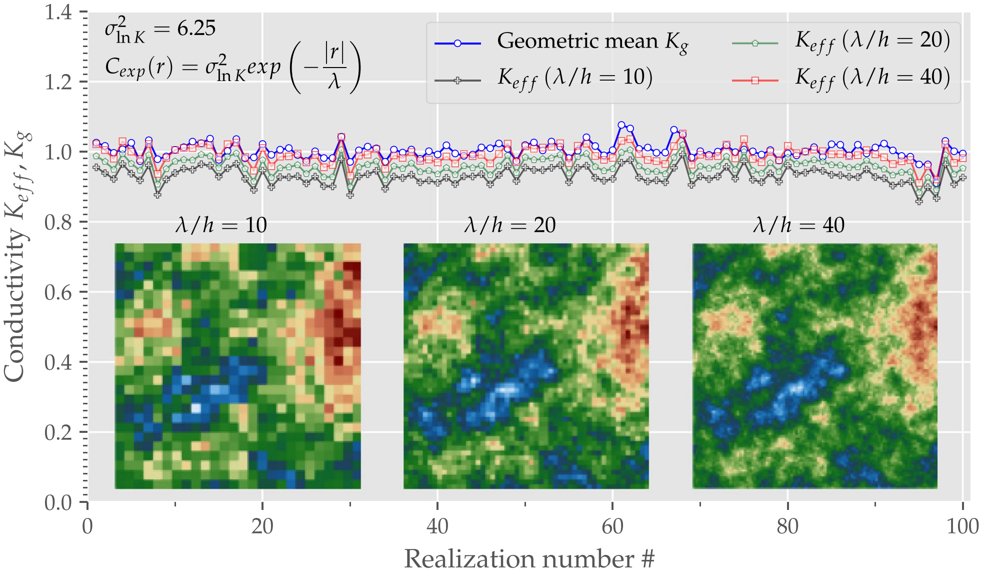


Fig. 8 Comparison of effective permeability (K_{eff}) defined as being the mean flux normalized by the total head gradient versus geometric mean of conductivity (K_g) for 100 different samples and three different resolutions. Domain of [410λ × 204λ], exponential covariance, variance $\sigma^2_{\ln K} = 6.25$, resolutions of λ/h =10, 20 and 40 ($h = \Delta x = \Delta y$), for clarity a [2.5λ × 2.5λ] inner window of one sample is displayed for each resolution



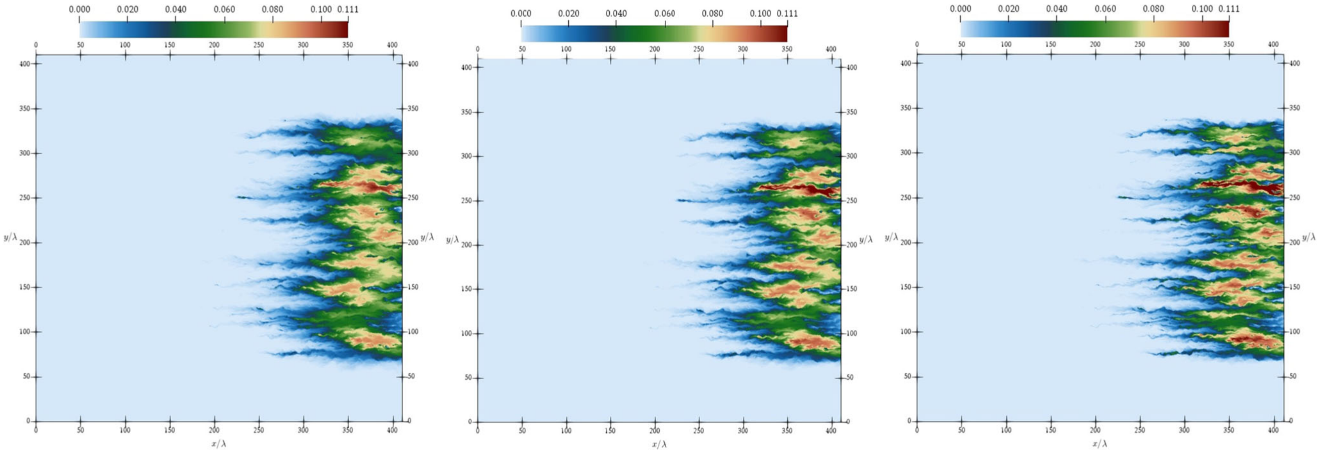


Fig. 9 Comparison for different types of transport $\sigma^2_{\ln K} = 2.25$ at final of simulation, from left to right: Aα, AD and PA

the conductivity field generated for variance $\sigma^2_{\ln K} = 6.25$ and the velocity fields calculated for different variances ($\sigma^2_{\ln K} = 1.00$ y $\sigma^2_{\ln K} = 6.25$).

Regarding the quality of the solution for the flow, Fig. 7 shows the comparison of the effective conductivity (K_{eff}) defined as the mean flow normalized by the head difference versus the geometric mean of the conductivity (K_{g}) [11]. Exponential and gaussian covariance models are compared for a variance of $\sigma^2_{\ln K} = 6.25$. A slight difference between K_{eff} and K_{g} can be seen for the exponential covariance model. This is due to the non-differentiability at 0 presented by the exponential covariance model [28].

Figure 8 compares the effective conductivity with the geometric mean for the exponential covariance model, for three different model resolutions. It can be noted that the difference between K_{eff} and K_{g} is reduced when solving for higher resolution because fluctuations are better captured. We highlight that the observed differences are due to an intrinsic problem of the exponential covariance model, as confirmed by Figs. 7 and 8.

4.3 Transport simulation

The following transport cases were considered in this work: pure advection (PA), advection-diffusion (AD) and isotropic advection-dispersion (Aα) with $\alpha_L = \alpha_T$, for the last two cases the Péclet numbers $Pe_d = \lambda u/D_m$ y $Pe_L = \lambda/\alpha_L$ are defined respectively. For the MC simulations a single value given by $Pe_d = Pe_L = Pe_T = 100$ is considered.

4.3.1 Dispersion computation

For the evaluation of the macrodispersion coefficient, the moments were calculated according to the following expression:

$$M_{nm}(t) = \iint_{\Omega} x^n y^m C(x,t)\,dx\,dy \qquad (29)$$

where $m + n$ is the moment order (con $m, n \in \{0; 1; 2\}$) and Ω the simulation domain. With this expression, the longitudinal and transversal macrodispersion coefficients are given by:

$$D_L(t) = \frac{1}{2\lambda \overline{u} N} \sum_{i=1}^{N} \frac{d}{dt}\left[\frac{M_{20}(t)}{M_{00}(t)} - \left(\frac{M_{10}(t)}{M_{00}(t)}\right)^2\right] \qquad (30)$$

$$D_T(t) = \frac{1}{2\lambda \overline{u} N} \sum_{i=1}^{N} \frac{d}{dt}\left[\frac{M_{02}(t)}{M_{00}(t)} - \left(\frac{M_{01}(t)}{M_{00}(t)}\right)^2\right] \qquad (31)$$

here $\overline{u}$ indicates the average velocity in the x direction of plume and $N = 100$ is the number of simulations for each parameter set.

4.3.2 Validation

As validation of the implemented Eulerian algorithms, results reported in the bibliography obtained by Lagrangian methods were reproduced. Figures 9 and 10 shows qualitatively the results for the same conductivity field for all the 3 transport cases (PA, AD and Aα).

Figure 11 compares the mean average longitudinal macrodispersion coefficients (over 100 realizations) for the AD and Aα cases for different $\sigma^2_{\ln K}$ with the results obtained by [6, 17].

4.3.3 Influence of variance on macrodispersion

Figures 12, 13 and 14 show results for the PA, AD, and Aα transport cases for different variance values. The asymptotic macrodispersion coefficients for each transport case are determined by averaging the coefficients of each of the 100

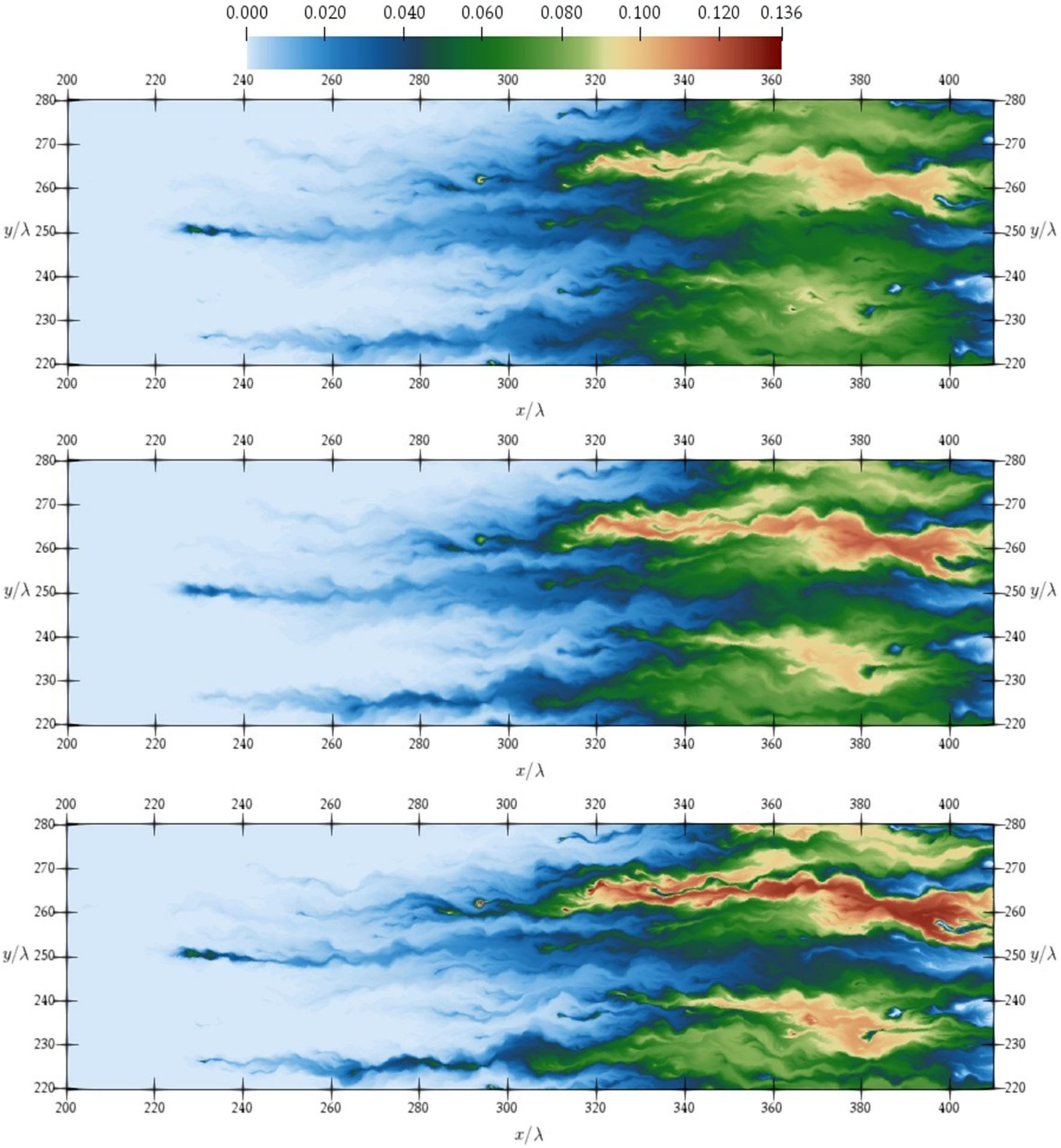


Fig. 10 Comparison for different types of transport $\sigma^2_{\ln K} = 2.25$ in a inner window of $[60x/\lambda \times 21x/\lambda]$, from top to bottom: A$\alpha$, AD and PA

simulations. For each simulation, the asymptotic value is calculated between the times $[t_0, t_1]$, where t_0 is the time in which there are no appreciable variations in the macro dispersion value, and t_1 the time before it begins to decay the solute mass in the domain.

In these figures it can be seen that for the different transport cases, the greater the variance, the greater the time in which the asymptotic value of the longitudinal macro dispersion is reached.

Figure 15 shows the times in which 95% of D_{LA} is reached for different variance values. It can be seen that the difference in the time in which the asymptotic value is reached, for the different transport cases (PA, AD and Aα), is not significant for variance values less than 2.5. However, for higher values, the difference in the time begins to be significant. This difference may be due to the fact that diffusion, and to a lesser extent dispersion, act as accelerators of the process of characterization of the medium. These processes make the solute to have a shorter residence time in areas with low conductivity, allowing the characterization of the macro dispersion to be carried out in a shorter time.

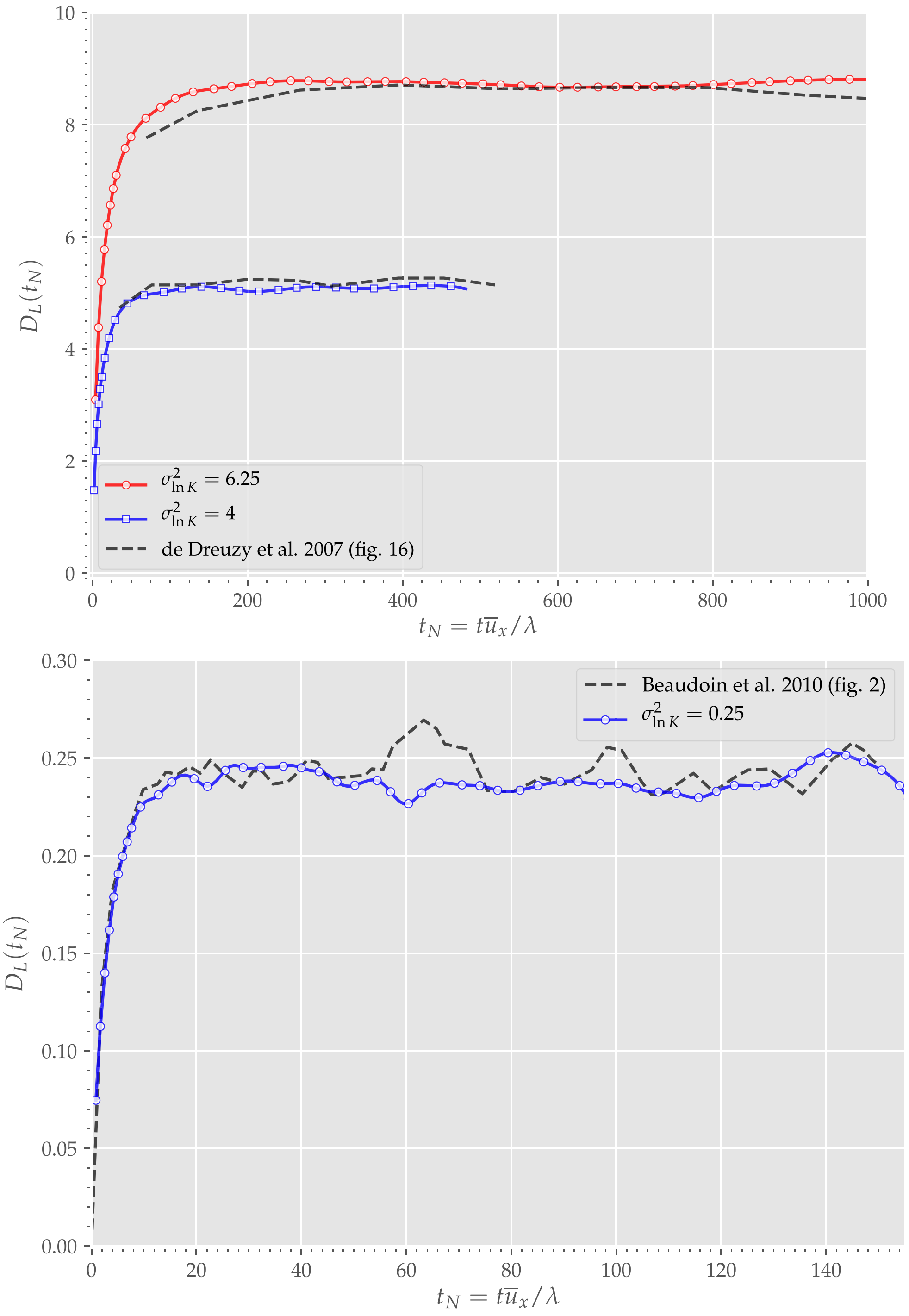


Fig. 11 Comparison for different types of transport, AD (top) and Aα (bottom), and variances $\sigma^2_{\ln K}$ with [6, 17]

4.3.4 Effects of diffusion and dispersion on macrodispersion

Figure 16 shows the asymptotic values for the PA case. It can be seen that the results are similar to those obtained in [17].

Figure 17 shows the asymptotic values of the longitudinal and transverse macrodispersion coefficients for the 3 transport cases (PA, AD and Aα) for different variance values. It can be seen that the difference in the asymptotic longitudinal macrodispersivity value, for the different

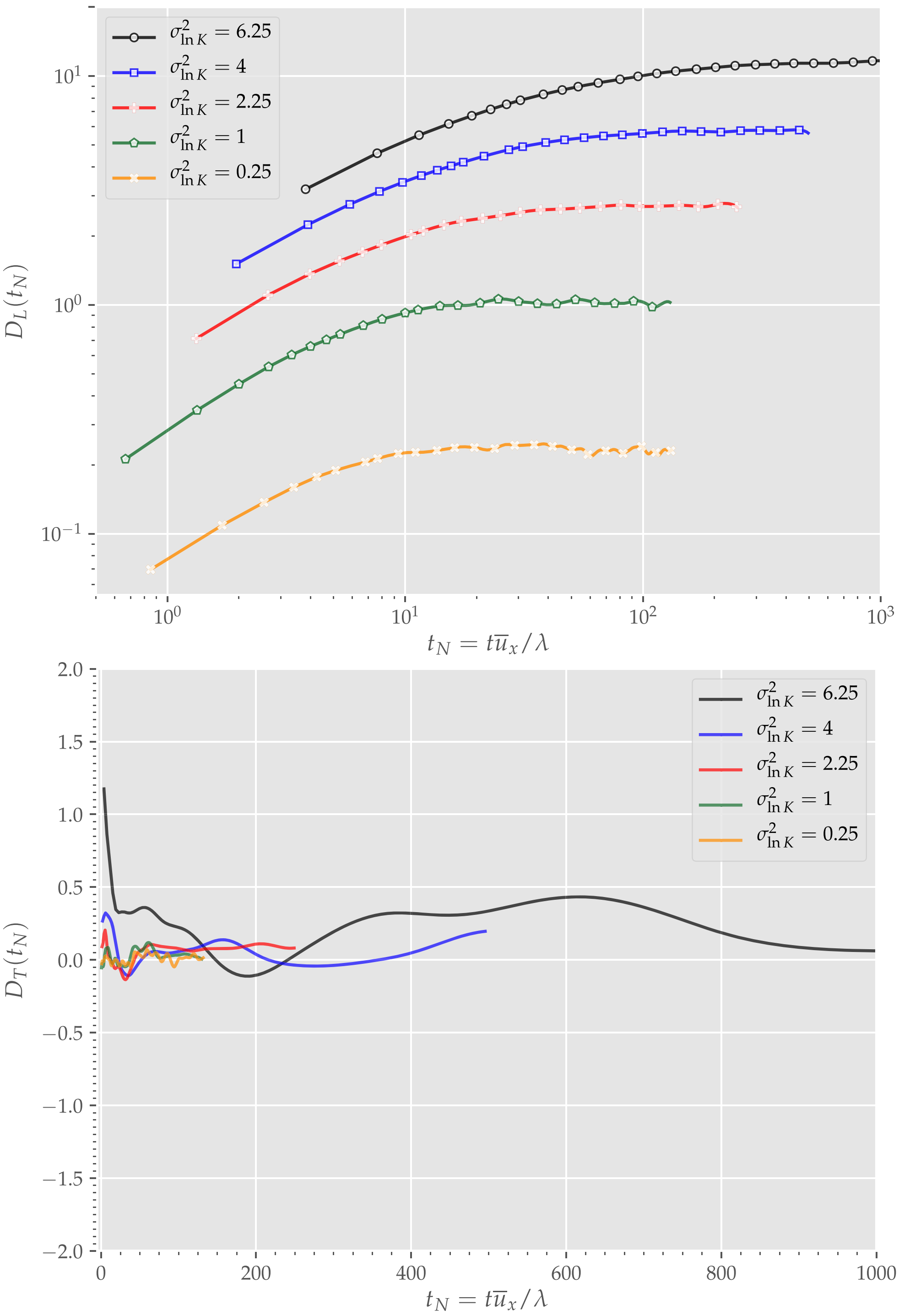


Fig. 12 Longitudinal and transverse macrodispersion for Pure Advection (PA) case

transport cases, is not significant for variance values less than 2.5. However, for higher values, the difference increases, which has been reported by several authors in the literature [6, 17, 39]. As these authors mention, diffusion and, to a lesser extent, dispersion, reduce the value of the macrodispersivity by reducing the residence time of the solute in areas of low conductivity. Note that the mechanisms that produce this effect are the same as those that affect the time for which the asymptotic value of longitudinal macrodispersivity is reached (shown on Fig. 15). Transverse dispersion on the other hand, is only weakly affected by the spatial heterogeneity and thus by

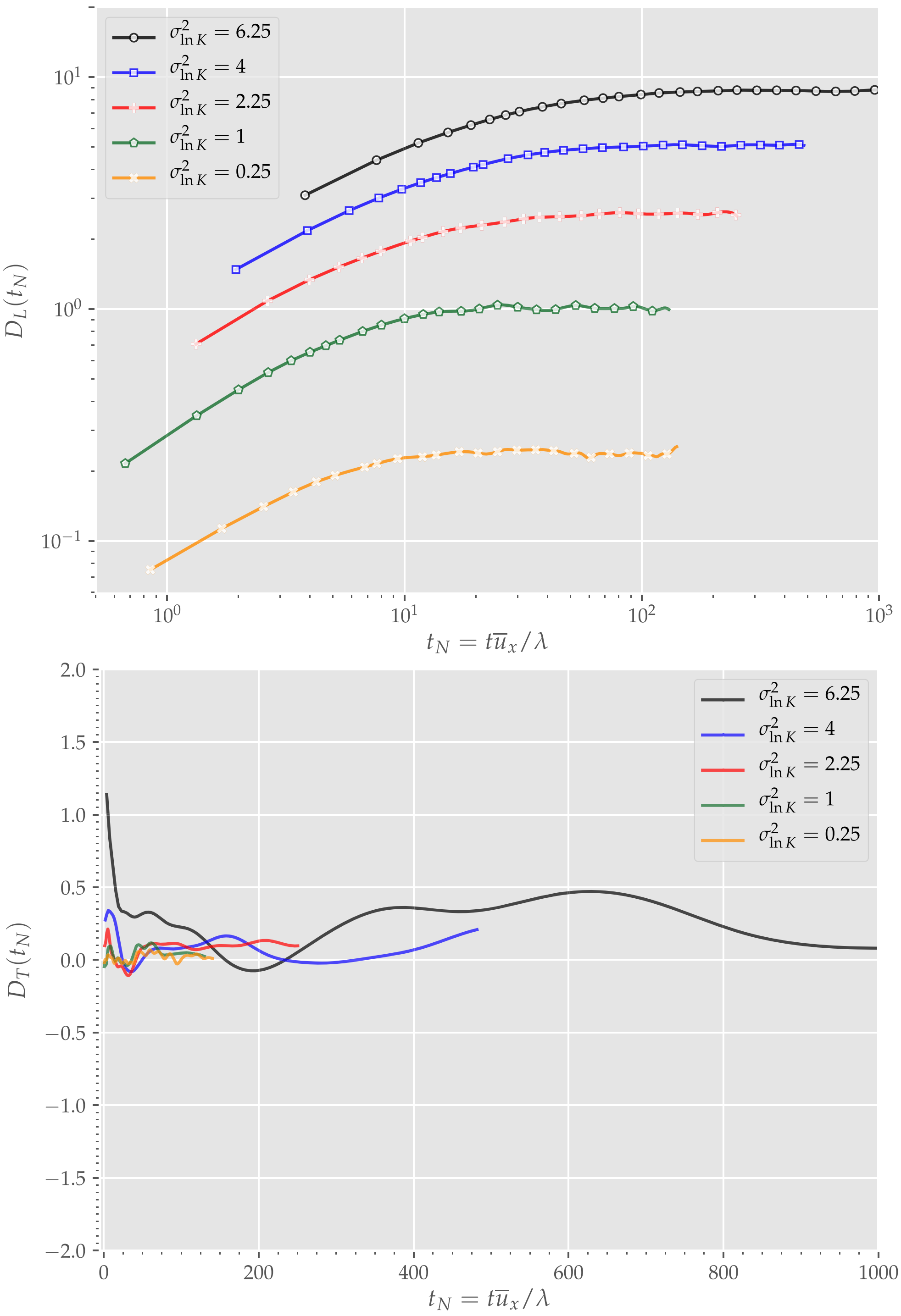


Fig. 13 Longitudinal and transverse macrodispersion for Advection-Diffusion (AD) case

the value of the variance of the log-conductivity. The fact that the velocity field is divergence-free impedes large transverse excursions of the streamlines and leads to a pure meandering. In fact, for the case of purely advective transport, transverse dispersion has been shown to be asymptotically zero, which is confirmed by the numerical simulations. For finite Péclet number, the transverse dispersion coefficients asymptote to a finite value, which however, tends to zero with increasing Péclet number [3]. The non-monotonic behavior for the transverse dispersion coefficients in Fig. 17 is rather a fluctuation than a systematic pattern, see also Fig. 16.

Fig. 14 Longitudinal and transverse macrodispersion for Advection-Dispersion (Aα) case

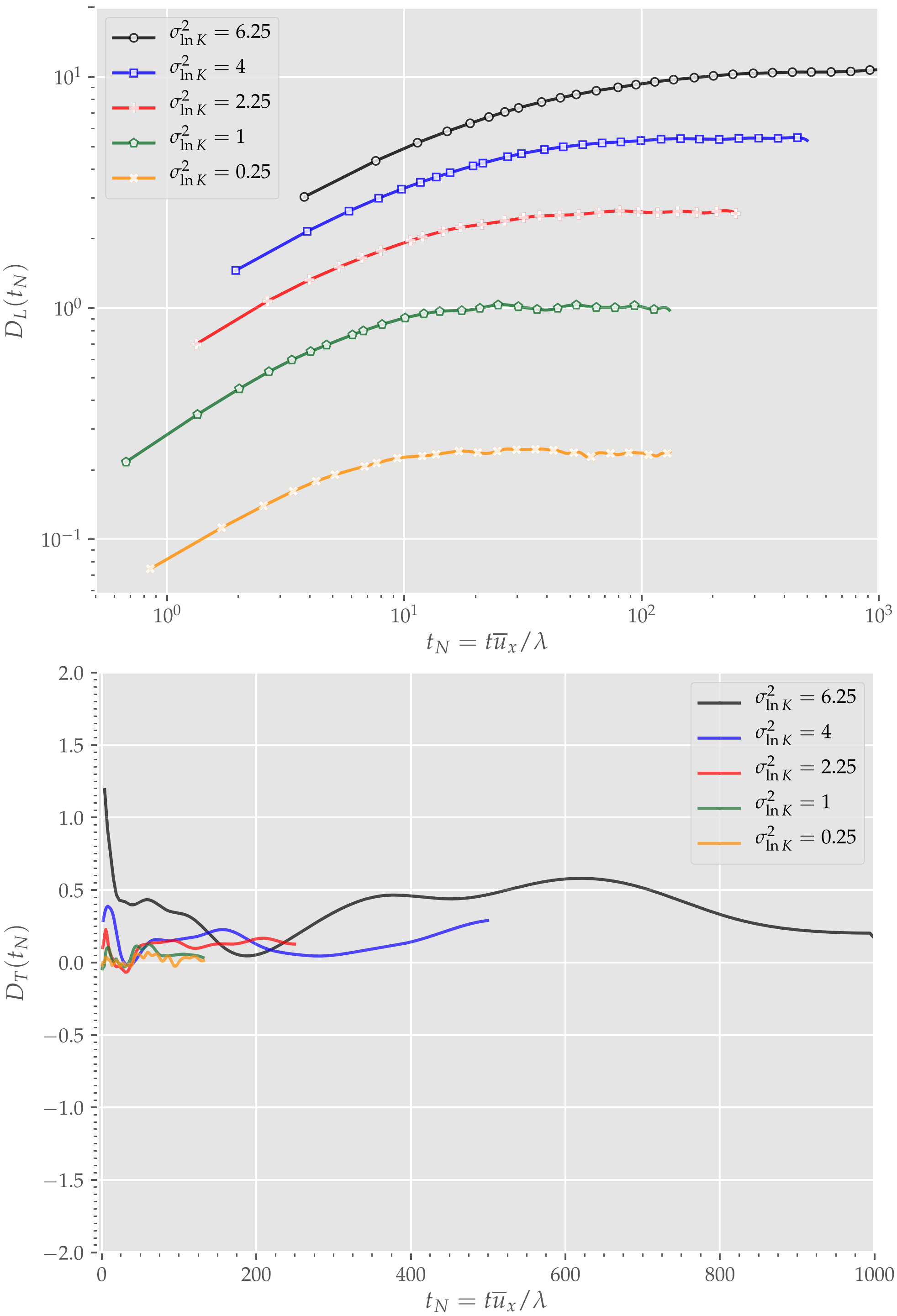


Table 3 compares the asymptotic macrodispersion values with respect to the case of pure advection, (D^{AD}_{LA}/D^{PA}_{LA} y $D^{A\alpha}_{LA}/D^{PA}_{LA}$) and it can be seen that the difference increases as $\sigma^2_{\ln K}$ grows and is negligible when $\sigma^2_{\ln K} < 1$.

The Fig. 18 shows for a case of $\sigma^2_{\ln K} = 4$ the coefficients D_L and D_T. In this case, the differences between the 3 transport cases on the asymptotic value of macrodispersivity and the time in which it is reached can be seen.

4.3.5 Performance of the implicit and explicit algorithms

The accuracy of the explicit and implicit methods using a TVD scheme was evaluated based on an error analysis

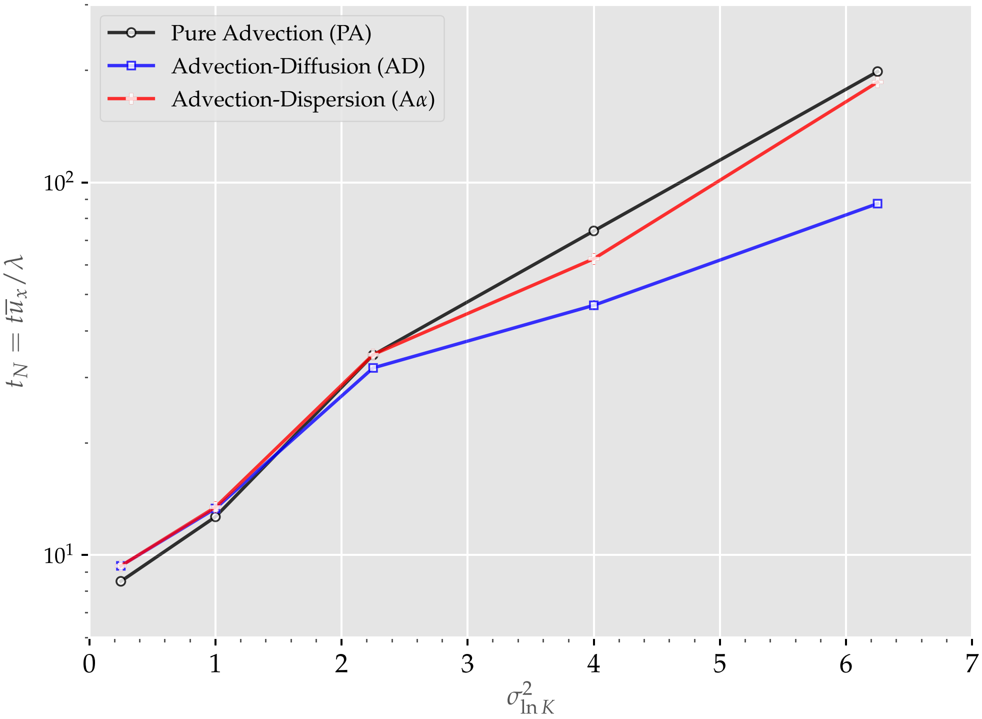


Fig. 15 Time scale to reach the asymptotic value for longitudinal dispersion as function of the variance of log conductivity

using a reference solution obtained by the implicit algorithm ($O(\Delta t^2)$) considering a very small time step. To make a fair comparison, a relationship between the time steps of each method was experimentally determined in order to obtain an error of the same order. Specifically, for the explicit method, the largest time step is used so that the solution remains stable throughout the simulation (maximum time from equation (17) times 1.5). For the implicit case, the time steps were chosen so that the error is of the same order as in the explicit case. The time steps used for the implicit method are between 8 and 34 times greater than those considered for the explicit method.

Figure 19 shows the evolution of the longitudinal and transverse macrodispersivities obtained with both methods for a particular case. As can be seen, both results are of the same order.

Table 4 shows the performance in terms of the computation time required for each transport case (average of 100 realizations) according to the type of simulated transport, the method used and the hardware where the simulation was run. It should be mentioned that for the explicit algorithm, the computation times for the PA and AD transport cases are numerically indistinguishable. This is due to the fact that the operations that are added in the AD case correspond to a multiplication using a fixed value (D_m) which does not imply global memory readings. For the Aα case more operations and global memory readings are added, in order to interpolate the dispersivity tensor at face centers, which significantly affects performance. The times reported in Table 5 shows that the explicit algorithm is much more efficient than the implicit algorithm in most cases, by a factor of up to 12. On the other hand, since in the implicit method a non-linear equation system must be solved for each time step, the total computation time is sensitive to the parameters set in the BiCGStab solver and the tolerance of the external loop for the deferred correction (DC). The type of transport that is simulated also affects performance, since the smoother the solutions (in the cases AD and Aα) the faster the convergence.

In Table 5 the ratios between implicit and explicit computing time for the different transport cases are compared. This table shows the effect that the size of the problem has on the methods performance for the three types of transport: the ratio shrinks as the problem grows. This is mainly because as the system grows, the numerical scheme considered for the implicit case becomes more efficient, due to the strategy used that links the BiCGStab solver and the deferred Correction approach. Since the result of the BiCGStab solver is used within an iterative loop, a maximum number of iterations was set instead of forcing the solver to reach a certain precision. This strategy can be dynamically adapted by monitoring the residue and increasing or decreasing the number of maximum iterations of the linear solver in each DC iteration.

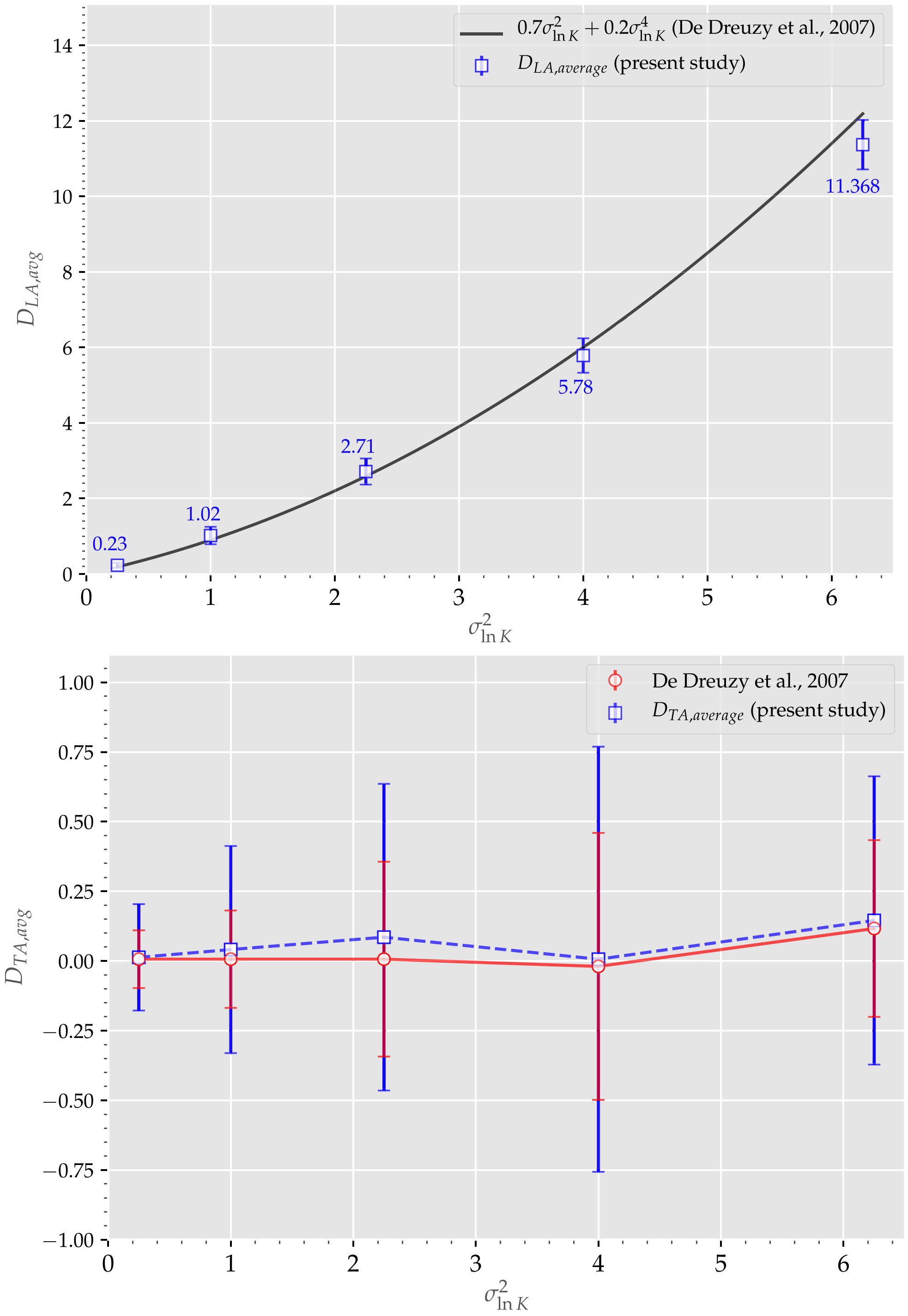


Fig. 16 Longitudinal and transverse asymptotic dispersion coefficients as function of log conductivity for pure advection. Vertical bars indicate standard deviation

Fig. 17 Longitudinal and transverse asymptotic dispersion coefficients as function of log conductivity for PA, AD and Aα transport mechanism

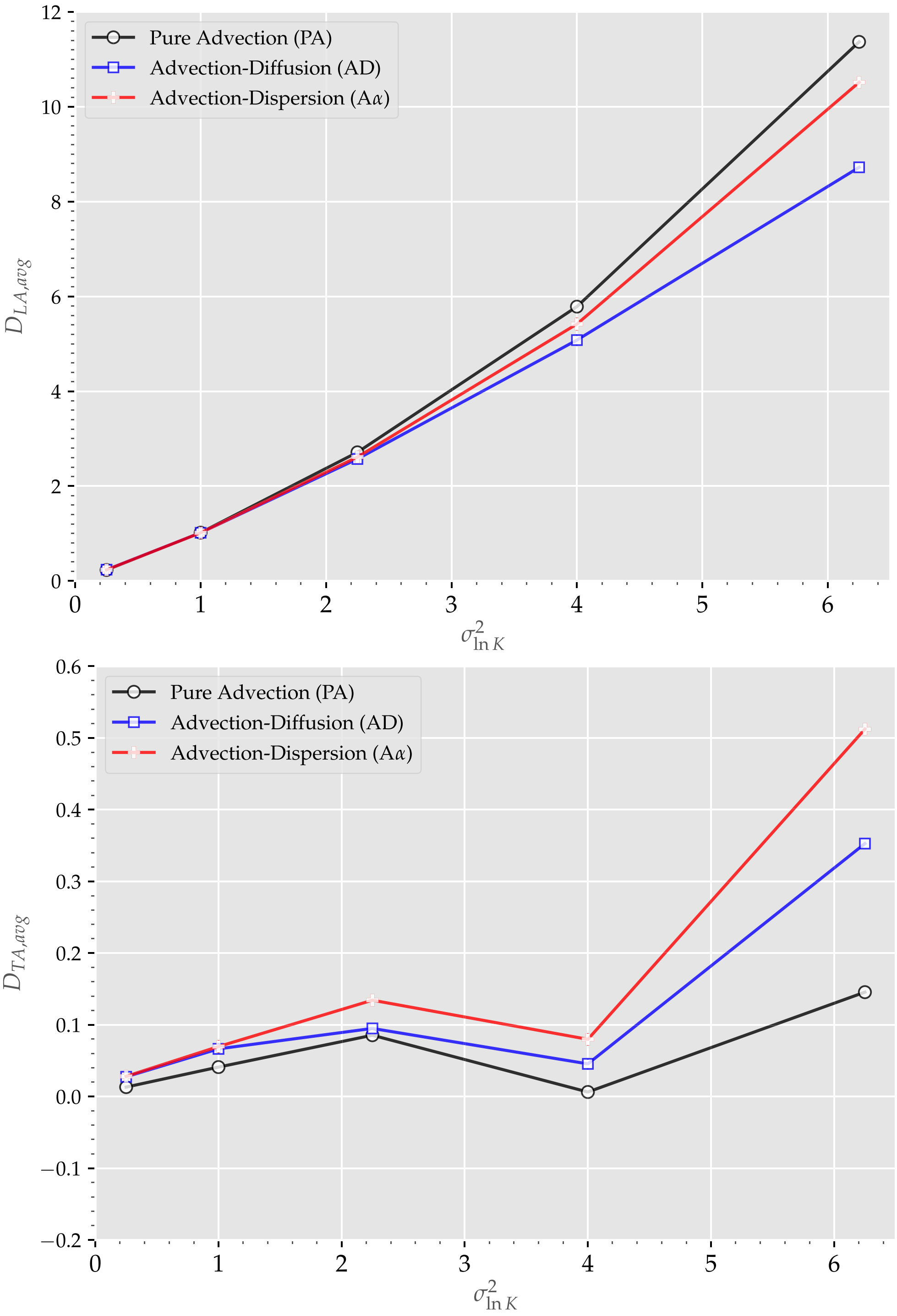


Table 3 Comparison asymptotic longitudinal macrodispersion for AD and Aα with PA case

	Advection-Difusion	Advection-Dispersion
Variance	$D_{LA}^{Pe_d=100}/D_{LA}^{PA}$	$D_{LA}^{Pe_L=Pe_T=100}/D_{LA}^{PA}$
0.25	1.030	1.020
1.00	0.997	0.996
2.25	0.948	0.915
4.00	0.879	0.936
6.25	0.767	0.925

It can also be observed that if the solution is smoother, as in the case of Aα, for the larger domain size the implicit method can be up to 2 times faster than the explicit one. On the other hand, the explicit method, despite of being slower than the implicit, has less memory requirement, which allows solving bigger problems. This is a particularly important aspect since the memory of GPUs is limited. As can be seen from Tables 4 and 5, the case of 134.48 million cells cannot be solved on the K40 GPU (the maximum size for implicit method is 107.08 million cells and 299.63 millions cells for explicit). For the V100 GPU the maximum size that can be solved is the one with 302.76 million cells for the implicit method and with 841 million cells for the explicit.

5 Conclusions

This work presents the implementation of two Eulerian methods in GPU to solve transport in dominant advective problems. For this, high resolution TVD schemes were implemented, resulting in algorithms with a spatial precision of $O(\Delta x^2)$. For the implicit algorithm, the TVD schemes were implemented using an iterative deferred correction approach. To the authors' knowledge, this is the first time that a fully Eulerian approach has been used to determine the macro dispersivity of highly heterogeneous media in a MC study context.

The algorithms were shown to correctly solve problems with sharp gradients for pure advection (PA), advection-diffusion (AD), and advection-dispersion (Aα), at values of $Pe = 100$ considering domains of up to 134.48 million cells. We considered the challenging problem of longitudinal and transverse macrodispersion for log-conductivity variances up to 6.25. Comparison with published numerical data obtained from high performance random walk particle tracking simulations validate the proposed numerical schemes. Furthermore, the numerical results shed some new light on the heterogeneity dependence of macrodispersion, and the time scales to reach the asymptotic behavior at finite Péclet numbers under local scale diffusion and dispersion.

The computation times show that the explicit method, despite being a $O(\Delta t)$ method and therefore requiring a shorter time step, has a better performance in GPU in most cases. This method, in addition to having a relatively simple algorithm to implement, requires less memory than the implicit one. The implicit method presented better performance for large problems. This behavior is due to the nonlinearity of the advective scheme used. This is mainly because as the system grows, the numerical scheme considered for the implicit case becomes more efficient, due to the strategy used that links the BiCGStab solver and the deferred Correction approach. The disadvantage of this method is that it requires more GPU memory, which on occasions can be a major limitation.

The computations times reported in this work show that it is feasible to carry out macrodispersion studies for highly heterogeneous media using fully Eulerian methods on GPU. This represents an attractive potential alternative for solving problems involving the interaction between the mobile and stationary phases, or the coupling between flow and transport, since the flow and transport calculations could be carried out on the same mesh.

Finally, we highlight that the implemented algorithms can be directly extended to solve 3D problems, using the parallel processing strategy via concurrent kernels. This can be done by using a similar kernel to process xy planes of the interior of the domain. This seeks to obtain a good balance between computations and the overhead introduced by the division and synchronization of the data, in addition to facilitating the implementation. In addition, the use of three-dimensional blocks is not recommended due to GPU block index limitations. Regarding the size of the domain for 3D, using the explicit algorithm in equipment 1, it is possible to solve domains of extensions up to 1280λ in streamwise direction and 25λ in the transverse directions with a resolution of $h = \lambda/10$. According to [5], with this dimensionalization, it is possible to simulate initial ergodic conditions and provide a correct sampling of the velocity field.

In conclusion, the presented fully Eulerian simulation methods provide an efficient alternative to particle-based simulators for transport in heterogeneous porous media. The Eulerian framework is more flexible for problems with complex boundary conditions, and problems with nonlinear feedback between flow and transport, such as in variable density or variable viscosity flow and transport problems.

Fig. 18 Comparison of normalized longitudinal and transverse dispersion coefficients as function of dimensionless time for all transport cases considering $\sigma^2_{\ln K} = 4$

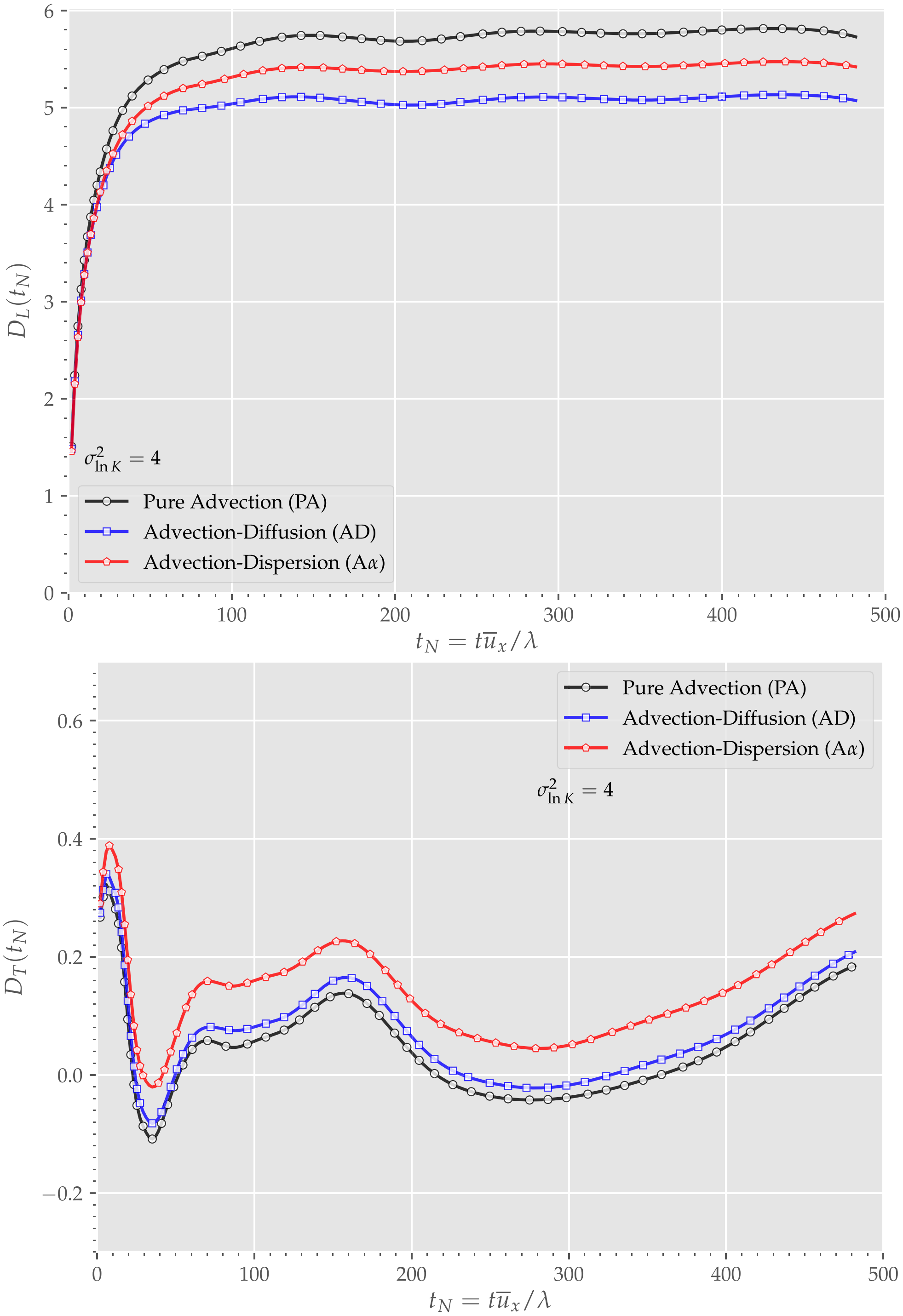


Fig. 19 Normalized longitudinal and transverse dispersion coefficient for single realization with $\sigma^2_{\ln K} = 4$, obtained using the explicit and implicit algorithm respectively

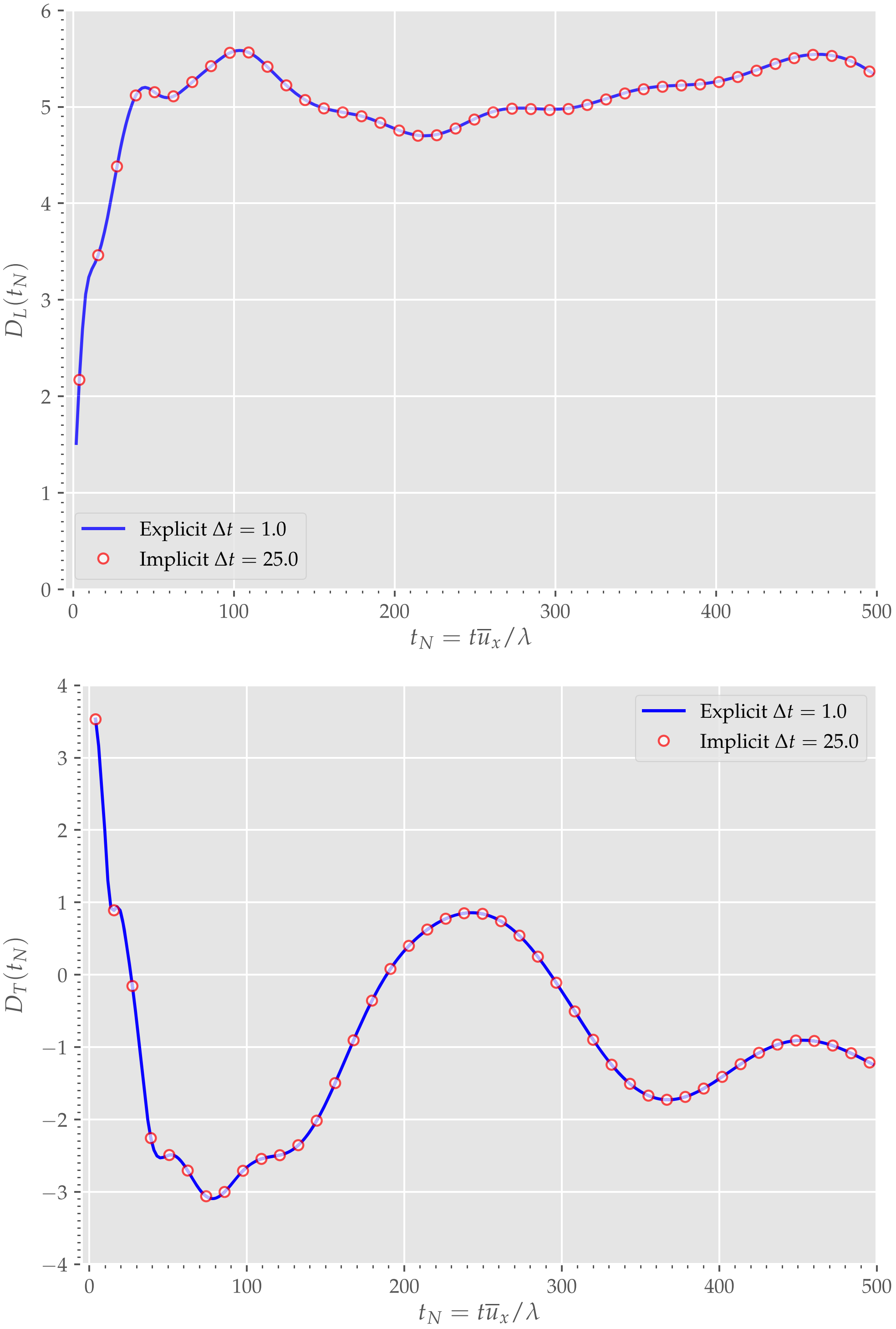


Table 4 Comparison of computing time for explicit and implicit methods for transport cases: Pure advection (PA), Advection-Diffusion (AD), Advection-Dispersion (Aα) running on different hardware

		$\sigma^2_{\ln K}$	0.25	1	2.25	4	6.25
		$N_X \times N_Y$ [Mcell]	4.198	4.198	16.81	67.24	134.48
		$t_{\text{final}}\overline{u}/\lambda$	190	190	350	600	1000
Equipment - GPU	Method	Transport case	Average computing time [min]				
Eq. 1 - Tesla K40	Explicit	PA / AD	0.68	1.52	18.5	368	2333
	Explicit	Aα	1.23	3.88	60,1	1046	6331
	Implicit	PA	5.98	7.68	66,6	571	*
	Implicit	AD	3.88	5.92	55,3	525	*
	Implicit	Aα	6.35	9.05	76,3	697	*
Eq. 2 - Tesla V100	Explicit	PA / AD	0.10	0.22	3,69	46,6	311
	Explicit	Aα	0.17	0.52	8,16	122	833
	Implicit	PA	1.22	1.57	12,3	101	409
	Implicit	AD	0.78	1.22	10,2	92,6	391
	Implicit	Aα	1.15	1.65	12,9	115	443

*Exceeds GPU memory of equipment 1

Table 5 Comparison of computing time for explicit and implicit methods for transport cases: Pure advection (PA), Advection-Diffusion (AD), Advection-Dispersion (Aα) running on different hardware

	$\sigma^2_{\ln K}$	0.25	1	2.25	4	6.25
	$N_X \times N_Y$ [Mcell]	4.198	4.198	16.81	67.24	134.48
	$t_{\text{final}}\overline{u}/\lambda$	190	190	350	600	1000
Equipment - GPU	Transport case	Ratio: $t_{\text{implicit}}/t_{\text{explicit}}$				
Eq. 1 - Tesla K40	PA	8.8	5.1	3.6	1.6	*
	AD	5.7	3.9	3.0	1.4	*
	Aα	5.1	2.3	1.3	0.7	*
Eq. 2 - Tesla V100	PA	11.8	7.0	3.3	2.2	1.3
	AD	7.6	5.4	2.8	2.0	1.3
	Aα	6.8	3.2	1.6	0.9	0.5

*Implicit method exceeds GPU memory of equipment 1

Acknowledgements L.B. acknowledges the support of UTN FRCON through "Scholarships for the training of doctors to strengthen the areas of R+D+i, Res. No. 1460/15".

Code Availability Some or all data, models, or custom code generated or used during the present work are available from the corresponding autor by request (C-CUDA program sources).

Declarations

Competing interests The authors declare that they have no conflict of interest.

References

1. Nvidia developer documentation: cuda toolkit documentation. http://docs.nvidia.com/cuda/cublas (2019)
2. Nvidia developer documentation: cuda toolkit documentation. http://docs.nvidia.com/cuda/curand (2019)
3. Attinger, S., Dentz, M., Kinzelbach, W.: Exact transverse macro dispersion coefficient for transport in heterogeneous media. Stoch Environ Res Risk Assess **18**, 9–15 (2004)
4. Bear, J.: Dynamics of Fuids in Porous Media. American Elsevier, New York (1972)
5. Beaudoin, A., de Dreuzy, J.R.: Numerical assessment of 3-d macrodispersion in heterogeneous porous media. Water Resour.

Res. **49**(5), 2489–2496 (2013). https://doi.org/10.1002/wrcr.20206

6. Beaudoin, A., de Dreuzy, J.R., Erhel, J.: Numerical monte carlo analysis of the influence of pore-scale dispersion on macrodispersion in 2-d heterogeneous porous media. Water Resources Research 46(12). https://doi.org/10.1029/2010WR009576 (2010)
7. Bellin, A., Salandin, P., Rinaldo, A.: Simulation of dispersion in heterogeneous porous formations: statistics, first-order theories, convergence of computations. Water Resour. Res. **28**(9), 2211–2227 (1992). https://doi.org/10.1029/92WR00578
8. Boso, F., Bellin, A., Dumbser, M.: Numerical simulations of solute transport in highly heterogeneous formations: a comparison of alternative numerical schemes. Advances in water resources **52**, 178–189 (2013). https://doi.org/10.1016/j.advwatres.2012.08.006
9. Chakravarthy, S., Osher, S.: High Resolution Applications of the Osher Upwind Scheme for the Euler Equations. In: 6Th Computational Fluid Dynamics Conference Danvers, pp. 1943 (1983). https://doi.org/10.2514/6.1983-1943
10. Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J.W., Skadron, K.: A performance study of general-purpose applications on graphics processors using cuda. Journal of parallel and distributed computing **68**(10), 1370–1380 (2008). https://doi.org/10.1016/j.jpdc.2008.05.014
11. Colecchio, I., Boschan, A., Otero, A.D., Noetinger, B.: On the multiscale characterization of effective hydraulic conductivity in random heterogeneous media: a historical survey and some new perspectives. Adv. Water Resour. **140**, 103594 (2020)
12. Comolli, A., Hakoun, V., Dentz, M.: Mechanisms, upscaling, and prediction of anomalous dispersion in heterogeneous porous media. Water Resour. Res. **55**(10), 8197–8222 (2019). https://doi.org/10.1029/2019WR024919
13. Crane, M.J., Blunt, M.J.: Streamline-based simulation of solute transport. Water Resour. Res. **35**(10), 3061–3078 (1999). https://doi.org/10.1029/1999wr900145
14. Cvetkovic, V., Cheng, H., Wen, X.H.: Analysis of nonlinear effects on tracer migration in heterogeneous aquifers using lagrangian travel time statistics. Water resources research **32**(6), 1671–1680 (1996). https://doi.org/10.1029/96WR00278
15. Dentz, M., Kinzelbach, H., Attinger, S., Kinzelbach, W.: Temporal behavior of a solute cloud in a heterogeneous porous medium 3. numerical simulations. Water resources research **38**(7), 23–1 (2002). https://doi.org/10.1029/2001WR000436
16. Domenico P.A., Schwartz FW (eds.): Physical and Chemical Hydrogeology. Wiley, NJ (1997)
17. de Dreuzy, J.R., Beaudoin, A., Erhel, J.: Asymptotic dispersion in 2d heterogeneous porous media determined by parallel numerical simulations. Water Resources Research 43(10). https://doi.org/10.1029/2006WR005394 (2007)
18. de Dreuzy, J.R., Carrera, J., Dentz, M., Le Borgne, T.: Time evolution of mixing in heterogeneous porous media. Water resources research 48(6). https://doi.org/10.1029/2011WR011360 (2012)
19. Fernández-Garcia, D., Sanchez-Vila, X.: Optimal reconstruction of concentrations, gradients and reaction rates from particle distributions. J. Contam. Hydrol. **120-121**, 99–114 (2011). https://doi.org/10.1016/j.jconhyd.2010.05.001
20. Godunov, S.K.: A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. Matematicheskii Sbornik **89**(3), 271–306 (1959)
21. Goode, D.J., Konikow, L.F.: Modification of a Method of Characteristics Solute Transport Model to Incorporate Decay and Equilibrium Controlled Sorption Or Ion Exchange 89-4030. Department of the Interior, US Geological Survey (1989)
22. Gotovac, H., Cvetkovic, V., Andricevic, R.: Flow and travel time statistics in highly heterogeneous porous media. Water resources research 45(7). https://doi.org/10.1029/2008WR007168 (2009)
23. Gupta R, Van Gijzen MB, Vuik CK: Efficient two-level preconditioned conjugate gradient method on the gpu. In: International Conference on High Performance Computing for Computational Science, Springer, pp 36–49 (2012). https://doi.org/10.1007/978-3-642-38718-0_7
24. Hakoun, V., Comolli, A., Dentz, M.: Upscaling and prediction of lagrangian velocity dynamics in heterogeneous porous media. Water Resour. Res. **55**(5), 3976–3996 (2019). https://doi.org/10.1029/2018WR023810
25. Harten A: High resolution schemes for hyperbolic conservation laws. Journal of Computational Physics **49**(3), 357–393 (1983). https://doi.org/https://doi.org/10.1016/0021-9991(83)90136-5
26. Harten, A.: On a class of high resolution total-variation-stable finite-difference schemes. SIAM J. Numer. Anal. **21**(1), 1–23 (1984). https://doi.org/10.1137/0721001
27. Hassan, A.E., Andricevic, R., Cvetkovic, V.: Evaluation of analytical solute discharge moments using numerical modeling in absolute and relative dispersion frameworks. Water resources research **38**(2), 1–1 (2002). https://doi.org/10.1029/2001WR000267
28. Hristopulos D.T.: Geometric properties of random fields. In: Random Fields for Spatial Data Modeling, Springer, pp 173–244 (2020)
29. Khosla, P., Rubin, S.: A diagonally dominant second-order accurate implicit scheme. Computers & Fluids **2**(2), 207–209 (1974). https://doi.org/10.1016/0045-7930(74)90014-0
30. Le Borgne, T., Dentz, M., Carrera, J.: Lagrangian statistical model for transport in highly heterogeneous velocity fields. Physical review letters **101**(9), 090601 (2008). https://doi.org/10.1103/PhysRevLett.101.090601
31. Le Borgne, T., Dentz, M., Davy, P., Bolster, D., Carrera, J., De Dreuzy, J.R., Bour, O.: Persistence of incomplete mixing: a key to anomalous transport. Physical Review E **84**(1), 015301 (2011). https://doi.org/10.1103/PhysRevE.84.015301
32. Leonard, B.P.: A stable and accurate convective modelling procedure based on quadratic upstream interpolation. Computer methods in applied mechanics and engineering **19**(1), 59–98 (1979). https://doi.org/10.1016/0045-7825(79)90034-3
33. Lindholm, E., Nickolls, J., Oberman, S., Montrym, J.: Nvidia tesla: a unified graphics and computing architecture. IEEE micro **28**(2), 39–55 (2008). https://doi.org/10.1109/MM.2008.31
34. Loppi, N., Witherden, F.D., Jameson, A., Vincent, P.E.: A high-order cross-platform incompressible navier–stokes solver via artificial compressibility with application to a turbulent jet. Comput. Phys. Commun. **233**, 193–205 (2018). https://doi.org/10.1016/j.cpc.2018.06.016
35. Moukalled F, Mangani L, Darwish M, et al: The finite volume method in computational fluid dynamics, vol 113. Springer (2016)
36. Niemi A, Bear J, Bensabat J (eds.): Geological Storage of CO2 in Deep Saline Formations. Springer Netherlands, Heidelberg (2017)
37. Noetinger, B., Roubinet, D., Russian, A., Le Borgne, T., Delay, F., Dentz, M., De Dreuzy, J.R., Gouze, P.: Random walk methods for modeling hydrodynamic transport in porous and fractured media from pore to reservoir scale. Transport in Porous Media pp 1–41 (2016)
38. Poinssot, C., Geckeis, H. (eds.): Radionuclide behavior in the natural environment: science, implications and lessons for the nuclear industry. Elsevier, Sawston (2012)
39. Ramasomanana, F., Younes, A., Ackerer, P.: Estimation of macrodispersion in 2-d highly heterogeneous porous media using the eulerian-lagrangian localized adjoint method. Water Resour. Res. **49**(1), 43–53 (2013). https://doi.org/10.1029/2012WR012228

40. Räss, L., Kolyukhin, D., Minakov, A.: Efficient parallel random field generator for large 3-d geophysical problems. Computers & Geosciences **131**, 158–169 (2019). https://doi.org/10.1016/j.cageo.2019.06.007
41. Rizzo, C.B., Nakano, A., de Barros, F.P.: Par2: Parallel random walk particle tracking method for solute transport in porous media. Comput. Phys. Commun. **239**, 265–271 (2019). https://doi.org/10.1016/j.cpc.2019.01.013
42. Roe, P.L.: Characteristic-based schemes for the euler equations. Annual review of fluid mechanics **18**(1), 337–365 (1986). https://doi.org/10.1146/annurev.fl.18.010186.002005
43. Ruan, F., McLaughlin, D.: An investigation of eulerian-lagrangian methods for solving heterogeneous advection-dominated transport problems. Water Resour. Res. **35**(8), 2359–2373 (1999). https://doi.org/10.1029/1999WR900049
44. Rubin, Y.: Stochastic modeling of macrodispersion in heterogeneous porous media. Water Resour. Res. **26**(1), 133–141 (1990). https://doi.org/10.1029/WR026i001p00133
45. Salandin, P., Fiorotto, V.: Solute transport in highly heterogeneous aquifers. Water resources research **34**(5), 949–961 (1998). https://doi.org/10.1029/98WR00219
46. Schwarze H, Jaekel U, Vereecken H: Estimation of macrodispersion by different approximation methods for flow and transport in randomly heterogeneous media. Transport in Porous Media **43**(2), 265–287 (2001). https://doi.org/10.1023/A:1010771123844
47. Shinozuka, M.: Simulation of multivariate and multidimensional random processes. J. Acoust. Soc. Am. **49**(1B), 357–368 (1971). https://doi.org/10.1121/1.1912338
48. Shinozuka, M., Jan, C.M.: Digital simulation of random processes and its applications. Journal of sound and vibration **25**(1), 111–128 (1972). https://doi.org/10.1016/0022-460X(72)90600-1
49. Sweby, P.K.: High resolution schemes using flux limiters for hyperbolic conservation laws. SIAM journal on numerical analysis **21**(5), 995–1011 (1984). https://doi.org/10.1137/0721062
50. Thibault J, Senocak I: Cuda implementation of a navier-stokes solver on multi-gpu desktop platforms for incompressible flows. In: 47th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition, p 758 (2009). https://doi.org/10.2514/6.2009-758
51. Toro EF: Riemann solvers and numerical methods for fluid dynamics: a practical introduction. Springer Science & Business Media (2013)
52. Trefry, M., Ruan, F., McLaughlin, D.: Numerical simulations of preasymptotic transport in heterogeneous porous media: Departures from the gaussian limit. Water Resources Research 39(3). https://doi.org/10.1029/2001WR001101 (2003)
53. Ujaldón M: Cuda achievements and gpu challenges ahead. In: International Conference on Articulated Motion and Deformable Objects, Springer, pp 207–217 (2016). https://doi.org/10.1007/978-3-319-41778-3_20
54. Van Leer, B.: Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. Journal of computational Physics **32**(1), 101–136 (1979). https://doi.org/10.1016/0021-9991(79)90145-1
55. Van der Vorst, H.A.: Bi-cgstab: a fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. SIAM Journal on scientific and Statistical Computing **13**(2), 631–644 (1992). https://doi.org/10.1137/0913035
56. Xu, J., Fu, H., Luk, W., Gan, L., Shi, W., Xue, W., Yang, C., Jiang, Y., He, C., Yang, G.: Optimizing finite volume method solvers on nvidia gpus. IEEE Transactions on Parallel and Distributed Systems **30**(12), 2790–2805 (2019). https://doi.org/10.1109/TPDS.2019.2926084
57. Younes, A., Ackerer, P., Lehmann, F.: A new efficient eulerian–lagrangian localized adjoint method for solving the advection–dispersion equation on unstructured meshes. Advances in water resources **29**(7), 1056–1074 (2006). https://doi.org/10.1016/j.advwatres.2005.09.003