Stochastic Modeling of Dispersion in Fractured Porous Media

Fabiano Sasselli

Bachelor of Science in Mechanical Engineering

Bachelor Thesis FS 20

Institute of Fluid Dynamics
ETH Zürich

Supervisor: Ranit Monga

Professor: Prof. Dr. Patrick Jenny

Abstract

Uncertainties in the characterization of fractured porous domains motivate the use of stochastic methods in the modeling of solute transport processes. Monga *et al.* 2019 have proposed a correlated random walk model for 2-D macrodispersion in random fractured media. In their data-driven approach for the stochastic model development, the ensemble of particle pathlines is generated using Monte Carlo simulation. They define effective particle pathlines which simplify the characterization of the correlations in particle's effective motion. The model is devised such that stochastic particle pathlines, generated with it, capture the correlations in the effective solute motion and also, fundamental features of the ensemble averaged solute concentration field. Some preliminary testing of the model has already been done by the authors for a limited number of fractured medium characterizations.

The goal of this thesis is to assess and extend the capability of the existing stochastic dispersion model developed at IFD by Monga *et al.* 2019. In this regard, reference Monte Carlo and stochastic simulations were performed on different medium characterizations which featured different fracture densities and log-conductivity variances for the heterogeneous matrix. The pathlines statistics resulting from the Monte Carlo and stochastic simulations were compared. It is shown that the proposed stochastic model is able to successfully capture the solute macrodispersion for the tested ranges of log-conductivity variances and fracture densities. Additionally, an alternative definition of effective particle pathline is proposed and implemented. It is a step towards eliminating the limitation of the current model in handling extremely low or extremely high fracture density scenarios. Furthermore, generation of fractures whose lengths follow exponential, log-normal and gamma distributions is implemented in the code.

Contents

1. Introduction **1**
1.1. Fractured porous media . . . 2
1.2. Fractures . . . 3
1.3. Fluid flow in porous media . . . 4
1.4. Dispersion in porous media I . . . 4

2. Stochastic computational methods **6**
2.1. Fractured porous field generation . . . 6
2.2. Velocity field . . . 6
2.3. Dispersion in porous media II . . . 7
2.4. Monte Carlo Simulation . . . 9
2.5. Stochastic Simulation . . . 10
2.5.1. Markov Process (MP) model . . . 10
2.5.2. Continuous-Time Random Walk (CTRW) model . . . 11
2.6. Stencil table representation of transition PDF: Transition Matrix . . . 12

3. IFD framework **14**
3.1. Monte Carlo Simulation . . . 14
3.1.1. Fractured porous field . . . 14
3.1.2. Darcy velocity solver . . . 16
3.1.3. Particle pathline tracking . . . 17
3.1.4. Monte Carlo fields realisations . . . 17
3.2. Stochastic Dispersion Model . . . 18
3.2.1. Particle pathline discretization . . . 18
3.2.2. Transition Matrices definition . . . 19
3.2.3. Transition Matrices generation . . . 20
3.2.4. Next state selection . . . 25
3.2.5. Pathlines generation . . . 26
3.2.6. SDM simulations settings . . . 26
3.3. Post-processing: statistics from MCS and SDM . . . 26
3.4. Already tested medium characterizations . . . 27
3.5. SDM limitations . . . 27
3.6. New tested medium characterizations . . . 29
3.7. IFD framework summarized . . . 32

4. Results **34**

5. Analysis **46**
5.1. Longitudinal mean $\langle x(\widetilde{t})\rangle$ . . . 46
5.2. Transverse mean $\langle y(\widetilde{t})\rangle$ . . . 46
5.3. Longitudinal σ_x^2 and transverse variance σ_y^2 . . . 46
5.3.1. Fracture density case: $\sigma_Y^2 = 1$, $f = 3\%$ . . . 46
5.3.2. Fracture density case: $\sigma_Y^2 = 1$, $f = 5\%$ . . . 46
5.3.3. Fracture density case: $\sigma_Y^2 = 1$, $f = 20\%$ . . . 46
5.3.4. Field variance case: $\sigma_Y^2 = 4$, $f = 5\%$: . . . 47
5.3.5. Field variance case: $\sigma_Y^2 = 1/16$, $f = 5\%$: . . . 47
5.4. Longitudinal $p(x(\widetilde{t}))$ and transverse $p(y(\widetilde{t}))$ position marginal PDF . . . 47
5.5. Fracture density effect on dispersion . . . 47
5.6. Fracture density effect on model accuracy . . . 48
5.7. Log-conductivity field variance effect on dispersion . . . 48
5.8. Log-conductivity field variance effect on model accuracy . . . 49
5.9. Asymptotic transverse dispersion . . . 50
5.10. SDM improvements . . . 50

6. Conclusions **55**

Bibliography **56**

A. Fractured porous fields **59**

B. IFD code framework improvements **66**

C. Field variance case: $\sigma_Y^2 = 1/16$, $f = 5\%$, $\theta_{fr} = 0°$ **72**

Attachements **74**

1. Introduction

The accurate, cost and time efficient simulation of flow and transport processes in fractured porous media is of high importance in many areas of engineering and natural sciences, such as: chemical engineering, filtration, soil and rock mechanics, petroleum engineering, construction engineering, hydrological engineering, geophysics, bioengineering, material science, battery technology [33], high level radioactive waste disposal, groundwater contamination and geothermal energy production, subsurface storage of materials fluids and energy sources, seawater intrusion in coastal aquifers, etc. [24].

Highly heterogeneous fractured porous media are characterized by a high degree of spatial variability of their properties over a vast range of length and time scales [20]; this leads to a lack of knowledge of the local properties and/or boundary conditions [6] and thus uncertainty in their value. The main interest towards fractures, comes from their ability to drastically speed up or hinder the transport process. Since in the models both the porous matrix and fractures properties, flow and transport need to be considered [12], this leads to too many uncertain parameters and uncertain results.

A solution to this uncertainty problem lies in the form of a stochastic description of the field parameters [20] which considers the uncertainties as random variables following a Probability Density Function (PDF). Another solution includes analysing many (thousands) porous fields realizations which have the same overall statistics, and then extract the statistical properties from all the transport processes of the different fields; this is the direct Monte Carlo Simulation (MCS) approach. Another improvement includes using some of the MCS data to create a simplified transport model, stochastic model, which depends only on a few relevant parameters of the fractured field and is statistically identical to the full MCS. This has the advantage of having a lower computational cost.

The framework developed at the Institute of Fluid Dynamics (IFD) focuses on subsurface fractured domains (underground soil/rock), where the interest is in predicting the macrodispersion of a solute (which will be also called contaminant plume) in the medium.
In the first part of this thesis the porous media flow and transport physics will be presented, afterwards fractured porous media will be characterized. In the second and third part, the simulation approaches and algorithms of the stochastic model implemented by IFD will be described. Finally the range of applicability is extended to different medium characterizations.

1.1. Fractured porous media

Fractured Porous media, as the name suggests, are porous media which have undergone a fracturing process and thus present fractures embedded in the porous matrix [8]. They are characterized by a vast size range of different types of interstitial openings which lead to a high heterogeneity of the medium, i.e. high spatial variability [8]. The medium is subdivided between primary pores, which are originated from the soil deposition process and even if small, occupy ca. 30% of the medium volume; and in secondary pores, also known as fractures, which constitute a very small percentage of the volume (1-3%) [8]. Due to the smaller size, primary pores have a lower permeability with respect to fractures, that on the other hand have (generally) large permeability values and thus act as fast flow and transport pathways [8]. This difference leads to fractures having a great impact in the fluid flow and transport process and the accurate modeling is imperative [8].

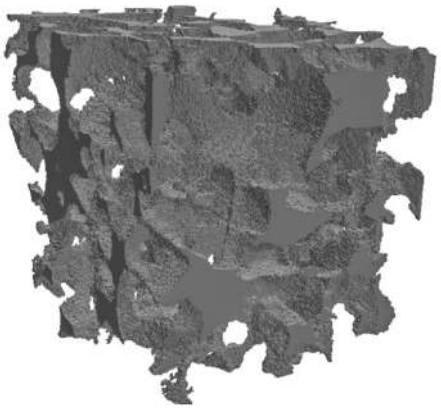


Figure 1.1.: Geometry image of Bentheimer Sandstone (porous medium). In gray is represented the porous volume separated by voids. Figure taken from Bijeljic *et al.* 2013 [36].

Properties of porous media are quantified by a set of parameters, the most relevant are permeability and hydraulic conductivity. The permeability k is a material property and quantify the ability of the medium to transmit fluids through it. A more permeable material let fluid flow with more ease, while an impermeable material does not allow fluid flow at all. The permeability is defined as:

$$k(\mathbf{x}) = K(\mathbf{x})\frac{\mu}{\rho g} \quad ; \quad [k] = m^2 \tag{1.1}$$

$K(\mathbf{x})$: hydraulic conductivity, m/s
μ: dynamic viscosity, $Pa.s$
ρ: fluid density, kg/m^3
g: gravitational acceleration, m/s^2

Natural (fractured) porous formations are neither spatially homogeneous nor uniform; there can be zones with ordered features while the neighbouring areas might have a completely disordered structure or even different medium characteristics. Due to the heterogeneous nature of the medium, local measurements are not representative for the whole interest area and, thus, a constant value for hydraulic conductivity cannot be defined for the whole domain. Instead it is possible (and more useful) to work with the hydraulic conductivity distribution [24]. Using a Probability Density Function (PDF) based description allow to define the expected value, standard deviation and variance of conductivity. Increasing the standard deviation of the hydraulic conductivity increases the spread of its values over the possible values range, leading to an higher number of extremely low and high permeability zones. [24].

Even if the medium is highly heterogeneous, it is not completely random. Field parameters as the log-hydraulic conductivity in neighbour areas share some common spatial dependence (not abrupt variation in medium characteristics). This structure arises from stratification of the medium, and is noticeable in the difference between transverse (vertical) and longitudinal (horizontal) correlation scales [7]. Using a stochastic approach, this spatial correlation can be well quantified by the PDF spatial covariance C_Y of the hydraulic conductivity [1,24].

1.2. Fractures

Due to the reorientation and stress displacement in the rock, fractures comes in different shapes and forms; these can be organized into families, each with its own set of parameters. Fractures have a great impact on the overall permeability of the medium. Single fractures can be idealized as very thin irregular channels, with their own surface roughness, width (also aperture), orientation with respect to the longitudinal axis (also orientation angle) and length [12]. The aperture is defined as the distance between the two surfaces of the fracture at each point along its length [10]. The fractures ensemble can be characterized by a fracture density; which is defined, for the considered two dimensional set-up, as the ratio of covered area by fractures to the total domain area. An increase in the fracture density can be caused by an increase in the size of fractures, number of fractures or both. More fractures leads to an higher connectivity in the medium. If different families of fractures (mainly with different orientation) are present, they can increase the connectivity in all directions [12]. On the other hand if only a single family of fractures is present (suppose all longitudinally oriented), the overlapping fractures will create a channel in one direction, leading to only one preferential path for fluid flow, surrounded by the slower porous path, and thus lower network connectivity [12]. Measurements show that the aperture is normally distributed for cases with small standard deviation, while for the cases with large standard deviation the distribution is log-normal [9]. The length distribution has been found to follow different distributions, precisely: exponential, log-normal, gamma and power law. Each of these distribution is suited for different fracture and rock systems [4].

The connectivity of the porous matrix has a big impact on the fluid flow as it can produce fast flow pathways or slow flow pathways; thus an increase in connectivity leads to an increase in the overall medium permeability. The path through open fractures has the highest hydraulic

conductivity; dispersion in a fractured system will thus happen primarily through the fractures [25]. Therefore in highly connected porous fields the fluid flow and transport depend less on the porous matrix and more on the fractures. The medium permeability is often anisotropic (direction dependent), and multiple fracture patterns affect the total flux of the system [12]. Fractures are not always empty, they can be partially or completely filled with a permeable (or not) material [5] and this is translated into two types of fractures: fractures with an higher permeability than the surrounding porous matrix and fracture with a lower permeability than the surrounding porous matrix [16]. In the latter the fractures act as obstacle for the fluid flow and transport.

1.3. Fluid flow in porous media

Fluid flow in porous media is well described by the Darcy law. This law derives from simplifications applied to the more general Navier-Stokes equation for incompressible fluids. The main assumptions are: incompressibility, stationarity and creeping flow ($Re \ll 1$) [31].
The general form of the flow rate (also volumetric flux, seepage velocity) is:

$$\mathbf{q}(\mathbf{x}) = -\frac{k(\mathbf{x})}{\mu}\nabla\mathbf{p} \quad ; \quad [\mathbf{q}(\mathbf{x})] = m/s \tag{1.2}$$

$k(\mathbf{x})$: medium permeability, m^2
μ: dynamic fluid viscosity, $Pa.s$
$\nabla\mathbf{p}$: pressure gradient (negative,also hydraulic head), Pa/m

This equation is coupled with the classic continuity equation for incompressible stationary flows [12]:

$$\nabla \cdot \mathbf{q}(\mathbf{x}) = 0 \tag{1.3}$$

From the Darcy formulation it is clear that the flow rate is directly proportional to the hydraulic conductivity:

$$\mathbf{q}(\mathbf{x}) \propto K(\mathbf{x})$$

Hence an uncertainty in the value of K leads to an uncertainty in the value of $\mathbf{q}$ and ultimately an uncertainty in the dispersed solute concentration since it is an advection controlled process. A way to account for these uncertainties is by considering the field parameters as random variables and use stochastic techniques, like Monte Carlo for data analysis and simulation.

1.4. Dispersion in porous media I

The term dispersion refers to the spreading of a solute through fluid into a porous medium. The dispersion has different scales: micro and macro scales. The microdispersion, also pore scale diffusion (PSD), is caused by diffusion and micro variations in flow velocity caused by the pores micro structure. Macrodispersion is mainly caused by the large scale medium heterogeneities

which are responsible for changing the flow direction (advection controlled) [15]. Macrodispersion is dominant for large Peclét numbers $Pe \gg 1$, where the dimensionless Pe number is an indication of the ratio of advection to PSD [3]. In general, higher the contrast in medium characteristics (i.e. hydraulic conductivity), higher the dispersion will be. This effect is due to the formation of high conductivity channels in the porous matrix. This higher dispersion effect can be seen when "the width of the velocity distribution is increased and the spatial correlation of the velocity field is stronger" as stated by Le Borgne *et al.* 2011 [15]. The dispersion is predominantly controlled by the large scale heterogeneity and less by the pore scale heterogeneity [7], therefore PSD can be neglected when modeling dispersion in such mediums. The scale dependant [2] dispersion in very heterogeneous fractured porous media is non-Fickian, due to the non-Gaussian spatial distributions and long tails of the dispersed solute plume [14], caused by mass transfer between fractures and porous matrix and contrast in flow velocities [35]. The conventional transport advection-dispersion model is not suitable to describe this behaviour [35] since it is not possible to define constant dispersion coefficients since they are scale dependent [21].

2. Stochastic computational methods

2.1. Fractured porous field generation

The stochastic approach to fractured porous fields characterization is suited for this application since a detailed description of the medium is not feasible [1]; instead the highly heterogeneous fractured porous medium is well described by a statistical heterogeneity characterization [14]. In this approach the hydraulic conductivity is considered as a random spatial variable which follows an appropriate PDF [24]. Each point of the domain has a (log-)hydraulic conductivity value depending on its probability distribution; the originating field is thus a unique realization of a spatial stochastic process [14]. Knowledge of the covariance function is very powerful, since in combination with the expected value of the PDF, it allows the complete description of the joint PDF (also multi variate distribution) of the hydraulic conductivity between all points in the domain, and thus a complete description of the field [1]. Generally to obtain asymptotic dispersion, a big computational domain is required [20,21].

As written in section 1.2, fractures are grouped into families; each family is characterized by the parameters: $\theta_{fr}, l_{fr}, a_{fr}$. They correspond to the inclination angle between fracture axis and longitudinal spatial direction, length in the direction of the main fracture axis and aperture of the fracture. Another important parameter to define when generating fractures is the density f of fractures, expressed by the percentage of covered area. To represent more accurately natural domains, it would be better to assign aperture and length values to different families using a user defined distribution. For example, exponentially distributed fractures lengths could be obtained and so on.

2.2. Velocity field

Once the field is generated and a computational grid is created on it, it is possible to simulate the fluid flow by solving the Darcy equation for the field with the appropriate Boundary Conditions (BCs) for velocity and pressure and initial conditions for non stationary problems; this will generate the random Eulerian velocity field [20]. There are multiple numerical schemes that can be used for the direct numerical simulation of the flow field. Eulerian velocity field refers to the utilization of an Eulerian frame of reference, i.e. a fixed control volume which allow to observe the velocity in every point in the domain. This is in contrast to the utilization of a Lagrangian description of the field, utilized for the transport modeling (see next section); in which the frame of reference moves in the flow field (attached to a fluid particle). Through the use of BCs a pressure gradient (hydraulic head) is applied to the whole domain. This gradient can be tilted with respect to the axes (x_1, x_2); this is useful to extend the number of possible simulated cases

in presence of fractures. Periodic BCs are particularly useful in Monte Carlo application due to their ability to represent a quasi-infinite domain [22a].

2.3. Dispersion in porous media II

As introduced before, the randomness of the velocity field also causes the particle position (path) to be random. Transport of a particle with volumetric concentration $C(x,t)$ is mainly caused by advection; for incompressible fluids the advection-dispersion Eulerian equation reads [17]:

$$\frac{D}{Dt}C(x,t) = \frac{\partial C}{\partial t} + u_i\frac{\partial C}{\partial x_i} = \frac{\partial}{\partial x_i}\left(D_{ij}\frac{\partial C}{\partial x_j}\right) \tag{2.1}$$

with dispersion tensor $\mathbf{D}$, which account for pore-scale dispersion. Neglecting pore scale dispersion (PSD) cancels the right hand side term [17]:

$$\frac{D}{Dt}C(x,t) = \frac{\partial C}{\partial t} + u_i\frac{\partial C}{\partial x_i} = 0 \tag{2.2}$$

Switching to a Lagrangian frame of reference; which follows a fluid particle (tracer), the particle travels along a pathline defined as [18]:

$$\mathbf{X}(t,\mathbf{X}_0) = \mathbf{X}_0 + \int_{t_0}^{t} u[\mathbf{X}(t',\mathbf{X}_0),t']dt' \tag{2.3}$$

$\mathbf{X}(t,\mathbf{X}_0) = (X_1,X_2)^T$: particle position (2-D) at time $t \geq 0$
$\mathbf{X}_0$: particle initial position at time t_0
t_0 : initial time

if PSD is neglected the concentration $C[x = \mathbf{X}(t,\mathbf{X}_0),t] \equiv C[\mathbf{X}_0,t]$ on the pathline is constant and the governing equation is [18]:

$$\frac{D}{Dt}C(\mathbf{X}_0,t) = \frac{\partial C}{\partial t} + \frac{\partial X_i}{\partial t}\frac{\partial C}{\partial x_i} = 0 \tag{2.4}$$

and with

$$\frac{\partial X_i(t,\mathbf{X}_0)}{\partial t} = u_i[\mathbf{X}(t,\mathbf{X}_0),t] \tag{2.5}$$

equations (2.2) and (2.4) are equivalent. The conclusion, as stated by Meyer *et al.* 2010 [18], is that "the Lagrangian representation of the concentration field based on tracer particles, is equivalent to the Eulerian description of the concentration field" [18]. Thereafter the concentration field is obtainable by tracking tracer particles, which are equivalent to fluid particles, in the flow field [21]. Due to the randomness of the velocity field also the concentration field become random.

The discrete motion of a particle in such random fields is described by [18]:

$$\mathbf{X}(t+\Delta t, \mathbf{X}_0) = \mathbf{X}(t, \mathbf{X}_0) + \underbrace{u[\mathbf{X}(t, \mathbf{X}_0), t]\Delta t}_{\Delta \mathbf{X}(\mathbf{X}_0)} \tag{2.6}$$

Δt: time interval

The spreading of the particle cloud, i.e. the particles ensemble, is quantified by the covariance tensor of the particles positions $\langle X_i' X_j' \rangle$, its evolution in time is described by [18]:

$$\frac{d^2 \langle X_i' X_j' \rangle}{dt^2} = 2\langle u_i'[X(\mathbf{X}_0, t), t] u_j'[X(\mathbf{X}_0, t), t] \rangle \tag{2.7}$$

From this result it is possible to derive the macrodispersion tensor, defined as [18] :

$$D_{ij} = \frac{1}{2}\frac{d}{dt}\langle X_i' X_j' \rangle \tag{2.8}$$

where $\mathbf{X}' = \mathbf{X} - \langle \mathbf{X} \rangle$ and $\mathbf{u}' = \mathbf{u} - \langle \mathbf{u} \rangle$ are the random fluctuation with respect to the mean value $\langle \cdot \rangle$.

The dispersion process can be modeled using the one time Eulerian tracer concentration PDF for a tracer injection, which describe the probability of having a certain concentration value $C(\mathbf{x}, t)$ [18]:

$$p_C(c; \mathbf{x}, t) = \delta(c - C_0) V_0 p_{\mathbf{X}}(\mathbf{x}; t) + \delta(c)[1 - V_0 p_{\mathbf{X}}(x, t)] \tag{2.9}$$

c : sample space coordinate of $C(\mathbf{x}, t)$
V_0: initial injection volume (for 2-D case become injection area)
C_0: initial volumetric concentration (for 2-D case become area concentration)
δ: Dirac Delta function
$p_C(c; \mathbf{x}, t)$: is the concentration field PDF with only argument c

Equation (2.9) is the PDF describing the random concentration field; from this PDF it is possible to derive the expected value $\langle C(\mathbf{x}, t) \rangle$ and the variance of the concentration σ_C^2, these can be easily found using the definition of the searched statistical quantities [18]:

$$\langle C(\mathbf{x}, t) \rangle = \int_{-\infty}^{\infty} c \cdot p_C(c; \mathbf{x}, t)\ dc = C_0 V_0 p_{\mathbf{X}}(\mathbf{x}; t) \tag{2.10}$$

$$\sigma_C^2 = \langle C(\mathbf{x}, t)^2 \rangle - \langle C(\mathbf{x}, t) \rangle^2 \tag{2.11}$$

The particle position PDF is described by [18]:

$$p_{\mathbf{X}}(\mathbf{x}; t) = \langle \delta(\mathbf{x} - \mathbf{X}(t, \mathbf{X}_0)) \rangle \tag{2.12}$$

$p_{\mathbf{X}}(\mathbf{x}; t)$: is the particle position PDF with only argument $\mathbf{x}$
$\mathbf{x}$: sample space coordinate of $\mathbf{X}$

and its expected position is:

$$\langle \mathbf{x} \rangle = \int_{-\infty}^{\infty} \mathbf{x} p_{\mathbf{X}}(\mathbf{x}; t)\ d\mathbf{x}$$

It is clear that there is a direct relation between the particle expected position and the first moment of the expected concentration. The relation between particle cloud spreading and position variance is shown better by the second spatial moment of solute plume:

$$I_C = \int_D \langle C(\mathbf{x}, t) \rangle (\mathbf{x} - \langle \mathbf{x} \rangle)^2 d\mathbf{x} = C_0 V_0 \sigma_{\mathbf{x}}^2$$

which give informations about how the concentration is distributed with respect to the axes of the domain D.

2.4. Monte Carlo Simulation

The name Monte Carlo (MC) refers to computational algorithms utilized for problems which present uncertainties whose solution is obtained by repeatedly generating random inputs from a PDF in the domain of interest; the obtained outputs ensemble statistics are then computed [27]. It is clear that this approach is very well suited for fracture porous media applications where the interest is in the macrodispersion behaviour. In these applications it is possible to analyse the statistical moments coming from a big number of realization of smaller domains or of a single realization of a large enough fractured porous domain so that the ensemble average can be approximated to the spatial average [23].

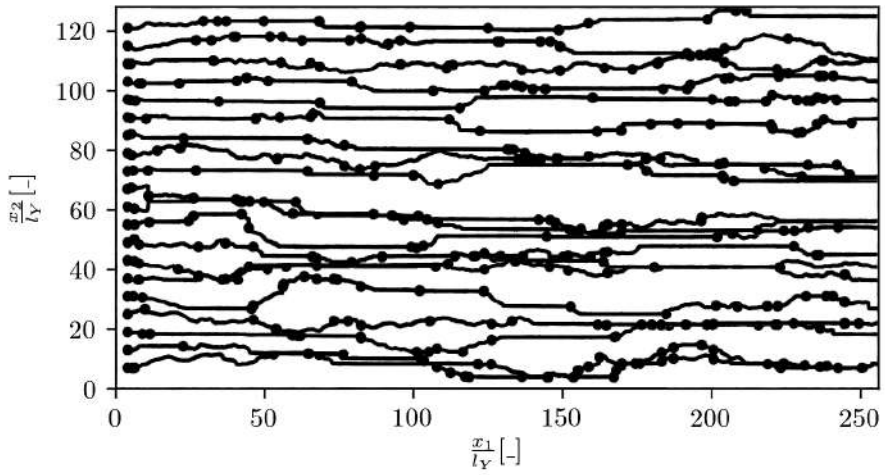


Figure 2.1.: Representation of 20 tracked pathlines (full black lines) on a single field realisation of a MC simulation. The black dots correspond to the delimitation of the effective pathlines segments created by the IFD model.

For fractured porous media application the MC scheme comprises of generating many random field realizations (hundreds, thousands) all based on the same stochastic model in order to have equivalent statistics. For each field realization, a fluid flow problem is solved with the appropriate BCs. At this point dispersion has to be simulated in the flow field in order to obtain the position data; one way to simulate dispersion includes tracking many fluid element (per field) Lagrangian motion as they were solute particles. With the obtained data it is now possible to compute the averaged statistics, i.e. mean and variance, of the particles cloud position.

Apart from the large number of required realizations, also a large number of particles are tracked in order to obtain more data (see Fig 2.1). In order to obtain uncorrelated pathlines in presence of fractures, the particles initial positions have to be spaced by a certain amount perpendicular to the mean pressure gradient [23]. It is possible to virtually enlarge the computational domain in each realization by using periodic BCs. Periodic BCs remove the influence of the domain boundaries [22a]. The main disadvantage of the MC simulation is the large number of realizations required in order to obtain converged statistics [6], this directly translate to an high computational cost even using fast solvers [6]. The required number of realisations needed to obtain accurate results also increases as the order of the searched statistical moment increases [22b]. To deal with these problems, stochastic modeling based simulations have been developed, with the computational cost advantage.

2.5. Stochastic Simulation

To decrease the computational cost of simulating dispersion in fractured porous media, it is possible to model the particle path in an efficient way. These models utilize only certain informations about the random particle paths in order to recreate them in a statistically equivalent way. Random Walk models based on Markov Processes are suited for this task.

2.5.1. Markov Process (MP) model

A MP model describes stochastic processes (succession of events, transition of states) where the probability of each event at a current state depends on the previous state alone [28]. To be modeled in such a way, processes must satisfy the Markov property [6]:

$$p(X_{n+1}|X_n, X_{n-1}, ..., X_1, X_0) = p(X_{n+1}|X_n) \tag{2.13}$$

X_n: random variable of stochastic process, n=0,1,...,N
n : step number of the stoch. process

in the case of time dependent variables, (2.13) states that the conditional PDF at the future (also next) state X_{n+1} at time $t_{n+1} = t_n + \Delta t$, depends only on the present state X_n at time t_n and no other past states [6,20] as introduced. For a stationary temporal MP process, the conditional PDF is the same for all states: $p(X_{n+1}|X_n) = const\ \forall n \geq 0$ [20], i.e. it does not depend on n and thus on the time.

Every hypothesized MP based model always has to be checked for its markovian nature. The Lagrangian motion of the tracer particles is hypothesised to be markovian, thus this hypothesis has to be checked. A method to do this, which consider three events X_{n+1}, X_n, X_{n-1} consist in first evaluating the Chapman-Kolmogorov equation [18]:

$$p(X_{n+1}|X_{n-1}; 2\Delta t) = \int_{-\infty}^{\infty} p(X_{n+1}|X_n; \Delta t)p(X_n|X_{n-1}; \Delta t)dX_n \tag{2.14}$$

and then compare the above with the conditional PDF $p(X_{n+1}|X_n)$ [6]. The conditional PDF are defined as:

$$p(X_{n+1}|X_n) = \frac{p(X_{n+1}, X_n)}{p(X_n)} \tag{2.15}$$

$p(X_{n+1}, X_n)$: joint PDF
$p(X_n)$: marginal PDF

A possible method to compute the conditional PDF is the Kernel Density Estimator algorithm (statistical numerical method) [20]. To be markovian, a process has to be independent from all past states but the immediately preceding one. A more precise way to check the markovian nature of a process consist in considering also the other past states, to do this multiple steps can be checked, i.e. [6]:

$$p(X_{n+1}|X_n, X_{n-k}) = p(X_{n+1}|X_n) \quad ; \quad k = 1, ..., n$$

For each step it is also necessary to check that the integral over the domain equal one and that the PDF is independent from the number of samples [6]. In the case of multiple random variables, i.e. X being a function of other random variables, it is necessary to ensure the statistical independence of the components of X before checking the markovian property [20]. The following is the notation that will be used in future sections of this thesis, the next state is defined with $j \equiv X_{n+1}$ and the present state is noted with $i \equiv X_n$. The probability of a transition from state i to state j is thus represented by $p(j|i)$.

2.5.2. Continuous-Time Random Walk (CTRW) model

Random Walk (RW) is an umbrella term for stochastic processes describing a discrete path (walk), where the position of each consecutive step is of random nature. These models are suited to model complex trajectories of apparent random nature or trajectories depending on highly uncertain parameters that can be assumed to be random [34]. This approach is thus well suited for the case of dispersion in porous media.

Contiuous-Time RW models are just a refined version of classic RW models, here the time interval between consecutive steps is random. The position and time interval are sampled from a joint PDF [20]. Since both the spatial and time steps are random, also the velocity between two points is random. CTRWs models applied to fractured porous media applications are especially

suited since no Fickian assumption has to be made [2] and they allow to incorporate the field heterogeneities through the Lagrangian velocities [11]. The particles trajectories through the domains are subdivided into steps; with each step having a certain probability of occurring in a defined time interval after the preceding step and moving to a new random position (a spatial step away) [3]. The goal of all the different CTRWs models is always finding the PDF governing the transitions [2].

Generally the step size (spatial) is constant for each step; but it is possible to have models where this is not the case. A version of CTRW with variable spatial steps is called generalized CTRW (also g-CTRW) [6]. Having variable step lengths is suited for particle motions in highly heterogeneous fractured porous fields due to the presence of different length scales (fractures, pores), where using the appropriate step length is important in order to consider all the medium features in an efficient way [6].

For Lagrangian particle motion in porous fields with log-normally distributed hydraulic conductivity, the walk has been found to be markovian, i.e. the Lagrangian velocities at equidistant positions along a particle pathline are correlated to each other and form a MP [13,14]; therefore it is possible to describe the particle state at each step using a MP. This means that the next state, after a constant spatial distance, depends on the current state. The correlated CTRW is a model that accounts for the markovianity of the velocities transitions [6]. Summarizing, CTRWs models (and variants) are used to simulate solute transport by governing the trajectories evolution of the tracer particles in the random field.

2.6. Stencil table representation of transition PDF: Transition Matrix

Transition matrices (TMs) are matrices used to describe the transition from a state i to a state j in a finite space Ω with cardinality (number of elements of the set) Ω [29]. From the definition of Markov Process, it follows that the elements of the TM are the conditional probability measures of the transition from state i (rows) to state j (column), noted $\mathbf{T}_{ij} = p(j|i)$ and also called "transition probabilities" [6].

$$\sum_{j=1}^{\Omega} \mathbf{T}_{ij} = 1 \quad \forall i \in [1, \Omega]$$

More specifically the rows represent the un-normalized conditional distribution functions of a state change $i \to j \; \forall \; j = 1, ..., \Omega$ [6], i.e. the entire i-th row represent the distribution of possible future states j that are obtainable from the current state i. The matrix has the form [29]:

$$\mathbf{T} = \begin{bmatrix} p(1|1) & \cdots & p(j|1) & \cdots & p(\Omega|1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ p(1|i) & \cdots & p(j|i) & \cdots & p(\Omega|i) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ p(1|\Omega) & \cdots & p(j|\Omega) & \cdots & p(\Omega|\Omega) \end{bmatrix}$$

In the case of fractured porous media the TM is defined by the data of the Lagrangian particle motion which is obtained using Monte Carlo techniques. Given the initial conditions, TMs can therefore be used to generate particles pathlines by finding the evolution of the state variables describing the particle pathlines using the conditional probabilities. The stencil table representation of the state transition PDF is very useful since it allows an easy the extraction of the next state index j [10]. Since the matrix cannot be infinitely large and the amount of data is finite, it is also necessary to "discretize" the Lagrangian data in order to create the TM. This data discretization consists of classify data of similar or equal value in groups using data bins. These data bins can be chosen to be of equal or different sizes depending on the application [14]. The bins dimensions are dependent on the number of "conditional" variables of interest, i.e. on the number of variables of the data joint space [6]. This discretization should produce a converged TM, i.e. the number of bins and data are large enough to represent accurately all transitions probabilities [6]. To compute the conditional PDFs, multiple methods are available; for example it is possible to use a stochastic differential equation (e.g. Langevin equation) to compute the PDFs [6] or use an Empirical Cumulative Distribution Function (ECDF) which utilizes the data to construct the Cumulative Distribution Function (CDF) and subsequently the PDF [20].

Starting from an initial state until the end state, every time the transition from $i \rightarrow j$, with current sate i and next state j, in the (joint) data spaces occurs, its overall transition frequency increases and therefore the probability of that particular state transition increases, i.e. the (i, j) element $\mathbf{T}_{ij}$ increase its value [10,20]. For example a particle going at a certain time from position state $i = 1$ with values $\mathbf{x}_1 = (x_1 = 20, y_1 = 5)$ to position state $j = 4$ with values $\mathbf{x}_4 = (x_4 = 32, y_4 = 6)$ will count as one occurrence contributing to $\mathbf{T}_{14}$. Assuming all the possible transition from $i = 1$ are $j \in \{1, 2, 3, 4\}$ (particle may also not move at all, that's why $j = 1$) with all an occurrence of one time except $j = 4$ occurring 3 times; then $\mathbf{T}_{14} = 0.5$ and $\mathbf{T}_{11} = \mathbf{T}_{12} = \mathbf{T}_{13} = 0.167$.

3. IFD framework

This chapter describes the IFD framework used in this thesis and the settings used in the simulations. In the previous chapters some references to the IFD framework were made, all of those will reappear in this more specific chapter. The goal is to obtain an overall understanding of how all the different algorithms work together to stochastically simulate new particle pathlines. Towards the end of the chapter the new cases that were simulated are presented and in the next chapter the results are shown.

3.1. Monte Carlo Simulation

3.1.1. Fractured porous field

The hydraulic conductivity is modeled to follow a log-normal distribution, therefore the parameter $Y(\mathbf{x}) = ln(K(\mathbf{x}))$ is more suited for the description [20]; in the considered framework the one-point bi-modal Gaussian PDF is used [21]:

$$p_Y(\gamma) = \underbrace{(1-f)\frac{1}{\sqrt{2\pi\sigma_Y^2}} \cdot \exp\left[-\frac{\gamma^2}{2\sigma_Y^2}\right]}_{\text{(I)}} + \underbrace{f\delta(\gamma - Y_{fr})}_{\text{(II)}} \tag{3.1}$$

σ_Y^2: variance of Y
$Y(\mathbf{x}) = ln(K(\mathbf{x}))$: log-hydraulic conductivity
Y_{fr}: log-hydraulic conductivity of fractures
f: fractional area covered by fractures (fracture density)
γ: sample space variable for Y
δ: Dirac delta function

The first mode (I) describes the distribution of effective log-conductivity Y in the porous matrix which also includes the effect of very small fractures. The second mode (II) describes the distribution of Y for large fractures [21].

For the covariance function an isotropic exponential function is proposed [21]:

$$C_Y(\mathbf{r}) = \sigma_Y^2 \cdot exp\left[-\frac{|\mathbf{r}|}{l_Y}\right] \tag{3.2}$$

l_Y : correlation length
$|\mathbf{r}|$: distance between two points

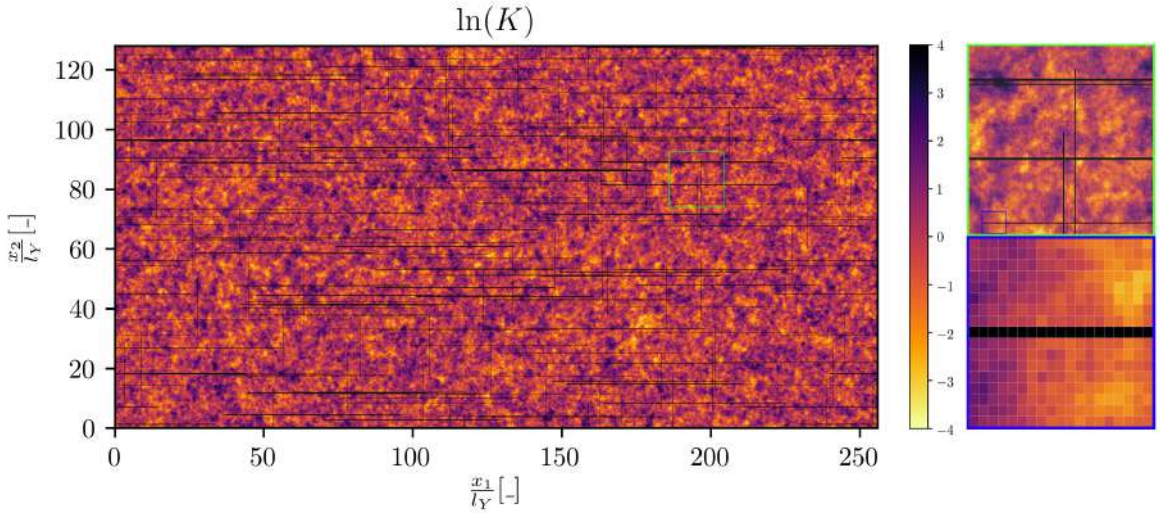


Figure 3.1.: Log-hydraulic conductivity field map generated with the IFD framework, 2 pixel line fractures families: $\theta_{fr} = 90°$, $l_{fr} = 20$, $a_{fr} = 0.0625$ and $\theta_{fr} = 0°$, $l_{fr} = 60$, $a_{fr} = 0.02$, the domain resolution is 4096× 2048 cells. The correlation length is 16 cells for both directions and $\sigma_Y^2 = 1$. In green (right-up) and blue (right-down) two zooms of a region of the field. The Fig. 1.1 can be thought as a single cell of this domain, with its own "effective/average conductivity" value.

The porous field is randomly generated following the bi-modal Gaussian log-hydraulic conductivity distribution and the exponential covariance function using a Fast Fourier Transform based approach. The dimensionless domain length in longitudinal direction is $l_x = x_1/l_Y = 256$, while in transverse direction is $l_y = x_2/l_Y = 128$. The whole domain is discretized with a resolution of 4096×2048 orthogonal cells. The correlation length l_Y is the same in both directions and is set to be 16 cells [21].

The fractures are also randomly generated in families, each characterized by the inclination angle with respect to the mean pressure gradient θ_{fr}, length l_{fr} and aperture a_{fr}. After setting the required density of fractures f (% covered area); the algorithm, following a uniform distribution, randomly populate the field with fractures of the different families until the desired coverage area (density) is reached. The constant log-hydraulic conductivity Y_{fr} of fractures is set to be a multiple of the variance square root (standard deviation) of Y, this means $Y_{fr} = w_{fr}\sigma_Y$ and the user has to choose an appropriate input weight w_{fr}. By varying the weight parameter it

is possible to obtain high or low conductivity fractures. The chosen $ln(K)$ distribution has zero mean $\langle Y \rangle$ and variance σ_Y^2. Similarly, for the porous matrix a weight for the multi-Gaussian background $w_{mg} = 1$ is set.

Fractures at an angle, with aperture of 1 cell, are represented (due to the orthogonal grid) as sequences of diagonal cells, connected just at their vertex, resulting in a disconnected fracture. The solution to this problem is to set the fractures at 0 or 90 degrees (longitudinal and transverse) and then change the mean pressure gradient angle with respect to the longitudinal direction.

All the features that influence the hydraulic conductivity but are smaller than the grid cell size (very small fractures, walls roughness, single pores, etc.) are not directly resolved, instead their effect is accounted by the log-conductivity PDF in the cell overall hydraulic conductivity. Thus the cell hydraulic conductivity value represent an average over the cell area.

3.1.2. Darcy velocity solver

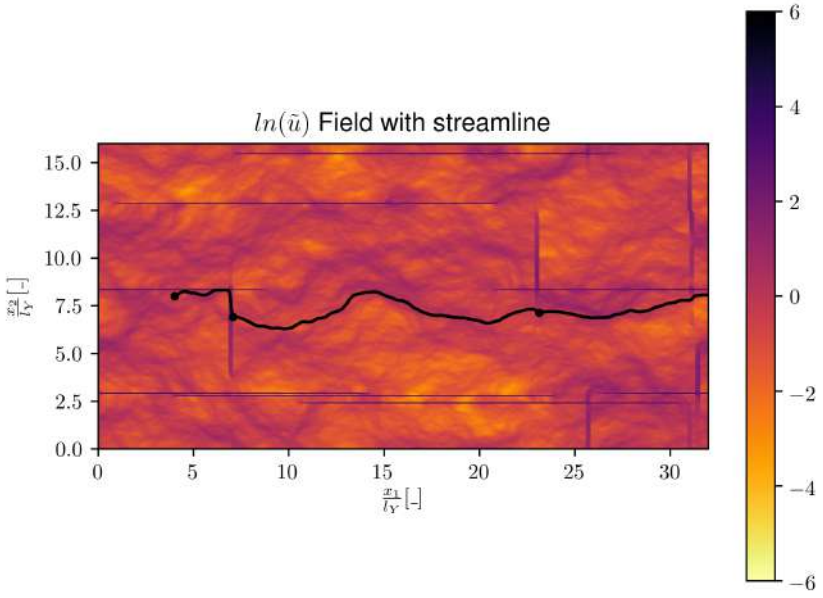


Figure 3.2.: Flow field representation with a tracked particle pathline (black line). Pixel lines fracture families: $\theta_{fr} = 0°$, $l_{fr} = 20$, $a_{fr} = 0.0625$ and $\theta_{fr} = 90°$, $l_{fr} = 6$, $a_{fr} = 0.25$, the domain resolution is 512×256 cells, size 32×16, fracture density 3%. Notice the particle pathline going along the regions with high velocity magnitude (darker shades). Logarithmic colorbar scaling for velocity magnitude.

The informations that the solver requires are: the mean pressure gradient between left and right boundary $\Delta h = 1$ (notation "h" due to the name hydraulic head), the ratio between the vertical and longitudinal pressure gradient components in order to tilt it h_2/h_1, the boundary condition type.

In this thesis a periodic BC is used, and the tolerance for the solution (max error) is $tol = 10^{-9}$. It is also possible to use other BCs: Neumann and Dirichlet. The pressure gradient Δh and velocity BCs are applied homogeneously from the left to the right boundaries. The solver uses these inputs in combination with the log-conductivity value of each cell to compute the flow and pressure fields.

In the considered framework, the solving algorithm consist of a combination of finite-volume and algebraic multi grid method [21]. A multi grid method is a multi resolution method, and is particularly suited for this application due to the difference in scales present in the medium. The algorithm allow to accelerate the convergence of the base method (here finite volume) by solving a coarse problem and then defining a global correction to apply to the finer grid problem [26]. Finite volume is used due to its conservative characteristic (mass conservation ensured).

3.1.3. Particle pathline tracking

The dispersion is modeled with the help of Lagrangian particle tracking in the Eulerian flow field [20]. Tracking is of fundamental importance in order to obtain the particle state at every point in the domain. The "tilde" notation indicates dimensionless quantities for position, time and velocity:

$$\widetilde{\mathbf{x}} = \frac{\mathbf{x}}{l_Y} = \frac{1}{l_Y}(x_1, x_2)^T = (x, y)^T \quad ; \quad \widetilde{t} = t\frac{U}{l_Y} \quad ; \quad \widetilde{u} = \frac{u}{U}$$

with mean flow velocity in longitudinal direction U. For a given random field realisation, a number n_{sl} of path lines (also streamlines) are tracked. The tracer particles are positioned at time $\widetilde{t} = 0$ at position $\widetilde{\mathbf{x}}_0 = (x = 4, y)$, i.e. at distance 4 from the left boundary. The y position is changed for every particle. Particles are uniformly distributed in transverse direction with an equal spacing. The number of released tracer particles depends on the fracture orientation: to avoid correlated pathlines (i.e. two particles start to follow the same path) the number of particles is 80 in case of $\theta_{fr} = 0°$, 12 in case of $\theta_{fr} = 90°$ and 32 if both fracture families are present. The particles pathlines data after 4 unit distance from the right boundary, i.e. $\widetilde{\mathbf{x}} = (x = l_x - 4, y)$, is not considered. The different particles tracked in each field do not interact with each other and therefore no pathlines intersections are present. The velocity data originated from the solver are given for cells interfaces, a linear interpolation in each direction is made to obtain the values at cells interiors [20].

3.1.4. Monte Carlo fields realisations

Monte Carlo simulations are performed in order to obtain a lot of useful data to test the Stochastic Dispersion Model (SDM). For each simulated case 800 random fields realization are considered.

The choice of the number of fields was done based on the consideration that the 1600 realization made by Monga *et al.* 2019 [21], were a conservative approach. After the simulation, both the high resolution pathline and the effective one are stored in order to be able to compare the transport statistics of the SDM to the reference MCS.

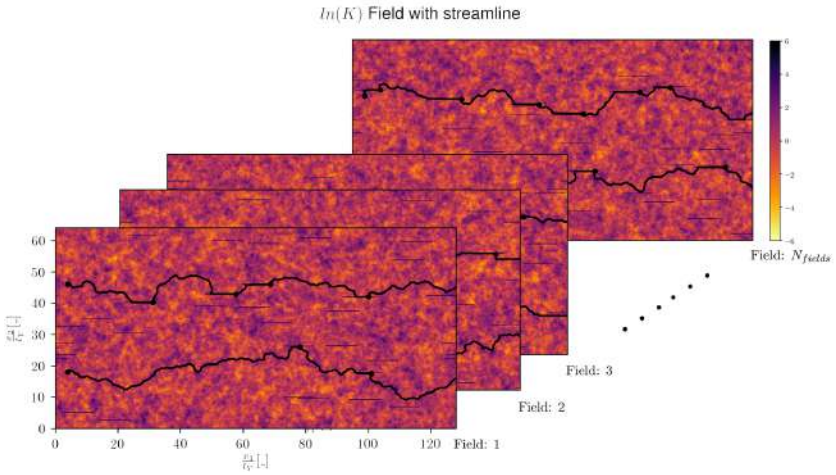


Figure 3.3.: Representation of Monte Carlo approach with four fields realisations. Two pathlines are tracked on each field (black line). The black dots correspond to the delimitation of the effective pathlines segments.

3.2. Stochastic Dispersion Model

3.2.1. Particle pathline discretization

The tracking originates high resolution pathlines which are then discretized into effective pathlines made of discrete linear segments, called effective segments [21]. In the adopted model, Monga *et al.* 2019 [21], defined the effective segments as "a line segment between two successive points of entry in the matrix post the travel in a fractured region" (the particle exit points from the fractures, also called effective data points). A general n-th effective segment, $n = 0, ..., N-1$ with N the total number of effective segments, is described by its length $\widetilde{l}_n$, direction inclination angle θ_n and average velocity $\widetilde{u}_n$ [21]. More precisely the described quantities are calculated as follow :

$$\theta_n = atan2(y_{n+1} - y_n, x_{n+1} - x_n) \tag{3.3}$$

$$\widetilde{l}_n = \|\widetilde{\mathbf{x}}_{n+1} - \widetilde{\mathbf{x}}_n\| = \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2} \tag{3.4}$$

$$\widetilde{u}_n = \frac{\widetilde{l}_n}{\widetilde{t}_{n+1} - \widetilde{t}_n} \tag{3.5}$$

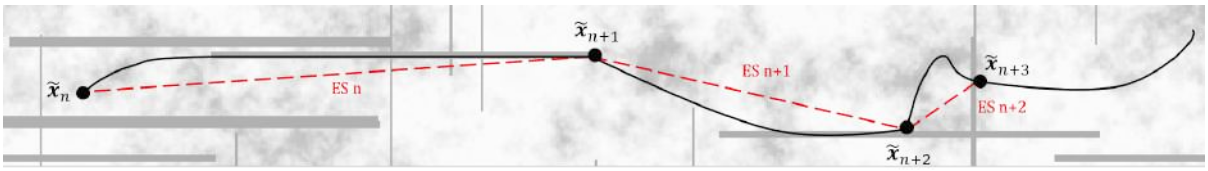


Figure 3.4.: Schematic representation of effective pathline (dashed red line) generation with respect to the high definition particle pathline (solid black line), in the background the porous matrix with fractures represented as thick grey lines. Effective Segments abbreviated with "ES".

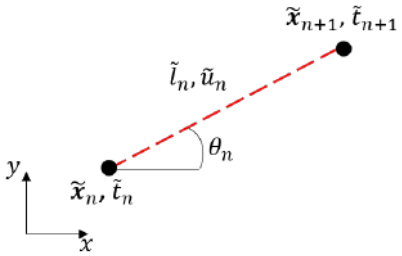


Figure 3.5.: Schematic representation of the state vector parameters computed for each effective segment (dashed red line) spanning between two effective data points (black dots).

For each n-th effective data point, two state vectors containing all the "effective data" are defined: one containing the current state data $\boldsymbol{\varphi'}_n = [\theta', ln(\widetilde{l'}), ln(\widetilde{u'}), y']_n^T \ \forall n = 0, ..., N-2$, i.e. of the current effective segment, and one containing the next state data $\boldsymbol{\varphi}_n = [\theta, ln(\widetilde{l}), ln(\widetilde{u})]_n^T \equiv \boldsymbol{\varphi'}_{n+1} \backslash \{y'\}_{n+1} \ \forall n = 1, ..., N-1$, i.e. of the next effective segment [21]. The vector components are denoted as ALU data (angle, length, velocity).

3.2.2. Transition Matrices definition

The stochastic model used for simulations developed at IFD is a data driven generalized correlated CTRW. It is a generalised random walk due to both spatial and temporal step length being randomly chosen, i.e. not of fixed length. The step length used by this correlated g-CTRW is equivalent to the geometrical effective segment since it is controlled by the segment length and angle random process. After each random step, a new state for the particle is defined based on the preceding one. The stochastic model is based on the following relations found by Monga *et al.* 2019 [21]:

$$ln(\widetilde{l}) = f(\theta', y') \ \ ; \ \ \theta = f(\theta', y') \ \ ; \ \ ln(\widetilde{u}) = f(ln(\widetilde{u})', \theta', ln(\widetilde{l})')$$

$f(\bullet)$: notation to represent variables statistical dependencies

i.e. in fields with transverse ($\theta_{fr} = 90°$) and longitudinal ($\theta_{fr} = 0°$) fractures, the θ process is correlated to the particle transverse coordinate y. The $ln(\widetilde{l})$ process is weakly correlated

with its preceding states values and there is some dependence between $ln(\widetilde{l})$ and $ln(\widetilde{u})$ (negative correlated). There is also dependence of $ln(\widetilde{l})$ on θ and in certain cases ($\theta_{fr} = 0°$) there is dependence between $ln(\widetilde{u})$ and θ [21]. Monga *et al.* 2019, found these relations by analysing the conditional PDFs obtainable from θ, $ln(\widetilde{l})$, $ln(\widetilde{u})$ and the correlation with their previous values for different fractures orientations. Therefore the whole state transition is regulated by two TMs [21]:

$$p(\theta, ln(\widetilde{l})|\theta', y') \quad ; \quad p(ln(\widetilde{u})|ln(\widetilde{u})', \theta, ln(\widetilde{l})) \tag{3.6}$$

One TM controls the angle and random walk step length while the other TM controls the velocity along the step. The transition conditional probabilities are stationary due to the log-conductivity fields also being stationary [21]. This model has been shown to capture the velocity statistics well enough.

Since the utilized TMs do not depend on all the variables present in $\boldsymbol{\varphi'}_n$, $\boldsymbol{\varphi}_n$, it is necessary to define new vectors which only contains the necessary data. Each TMs has its own set of modified state vectors, for the angle and segment length TM this leads to the definition of:

$$\boldsymbol{\phi'} = [y', \theta']^T \subset \boldsymbol{\varphi'} \quad ; \quad \boldsymbol{\phi} = [\theta, ln(\widetilde{l})]^T \subset \boldsymbol{\varphi}$$

while for the velocity TM:

$$\boldsymbol{\phi'} = [ln(\widetilde{u}'), \theta, ln(\widetilde{l})]^T \subset (\boldsymbol{\varphi'} \cup \boldsymbol{\varphi}) \quad ; \quad \boldsymbol{\phi} = [ln(\widetilde{u})]^T \subset \boldsymbol{\varphi}$$

From these modified state vectors, for each TM, the respective joint data spaces are defined:

$$\boldsymbol{\Phi'} = \bigcup_{n=0}^{N-2} \boldsymbol{\phi'}_n \quad ; \quad \boldsymbol{\Phi} = \bigcup_{n=0}^{N-2} \boldsymbol{\phi}_n$$

These joint spaces contains the data of the states used in the Transition Matrix. For the initial conditions an apposite TM is used, therefore apposite joint spaces are needed:

$$\boldsymbol{\Phi}_0 = (\theta_0, ln(\widetilde{l}_0)) \quad ; \quad \boldsymbol{\Phi}_0 = (ln(\widetilde{u}_0))$$

The transition from a current state i to a successive state j is controlled by the conditional transition probability $p(\boldsymbol{\phi}_j|\boldsymbol{\phi}'_i)$, and the ensemble of possible transitions is described by the Transition Matrix [21].

3.2.3. Transition Matrices generation

The obtained data is now used to construct the Transition Matrices that characterize the state change. The TMs construction and utilization process can be subdivided into the following steps: 1) creation of state vectors and modified state vectors using effective data (effective segments) from MCS, 2) creation of the joint spaces containing the realizations (data) of the modified state

vectors, 3) bins creation (boundaries and indices) for each dimension, i.e. random variable, in each joint space and data classification, 4) linearization of bins indices to i, j notation for the joint spaces of current state and future state respectively, 5) count each $i \rightarrow j$ transition to obtain $\mathbf{T}_{ij} \equiv p(j|i)$ and finally 6) generate pathlines using the TMs to find the new state of the RW.

The first and second step follow the methodology previously explained. The binning process consists in grouping data into containers. In the used model the data is taken from all the modified state vectors $\boldsymbol{\phi}$,$\boldsymbol{\phi'}$, whose ensemble create therefore multidimensional joint spaces (sets) notated $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi'}$ respectively. The name containers/bins refers to bounded regions in the data space; the data present in that region is stored in a defined bin at which an index is then assigned which describes a specific state [6]. There are several ways to subdivide $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi'}$; the approach used in this framework includes "a recursive usage of a binning scheme which make use of the marginal Empirical Cumulative Distribution Function (ECDF)" and classify the data into equi-probable bins [21]. The idea of the ECDF approach is to utilize the data distributions to obtain bins of different sizes with all containing approximately the same number of data [6] (approximately uniform distribution of data points [20]); this is made to "counter" the non uniformity of the data distribution and obtaining therefore a more balanced representation of all possible states (otherwise certain data points, i.e. states, would be incorporated into other bins instead of having their own bin) [6].

For each dimension d of $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi'}$, which correspond to the number of random variables of the space, an ECDF for the random variable $X^{(d)}$ (which could represent any of the d variables of the joint space) is determined as follows [21,30]:

$$F_{X^{(d)}}(x^{(d)}) = \frac{1}{n_{X^{(d)}}} \sum_{i=1}^{n_{X^{(d)}}} \mathbf{1}_{X_i^{(d)} \leq x^{(d)}} \tag{3.7}$$

$n_{X^{(d)}}$: maximum number of realisations of random variable $X^{(d)}$
$x^{(d)}$: coordinate in dimension d
$\mathbf{1}_A$: event indicator of event A, 1 if A occurs, 0 otherwise

The bin boundaries (noted with $\hat{\bullet}$) are linearly distributed in [0,1] in the space $F_{X^{(d)}}(x^{(d)})$ [21]:

$$\hat{x}_l^{(d)} = \frac{l}{n_b} \quad \forall \quad l \in \mathcal{Z} : 0 \leq l \leq n_b \tag{3.8}$$

n_b: number of classification bins

The bins boundary values are:

$$x_l^{(d)} = F_{X^{(d)}}^{-1}(\hat{x}_l^{(d)}) \tag{3.9}$$

The recursive part of the approach means that after classifying one dimension (X in Fig 3.6), the classification is recursively done on the other dimensions, i.e. the bins boundaries for the remaining dimensions (i.e. random variables) are obtained by considering the data distribution inside each one of the already defined bins until the maximal dimension is reached [20].

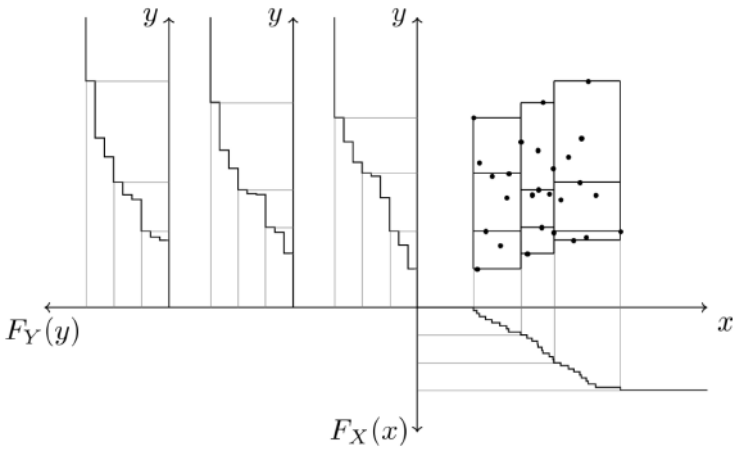


Figure 3.6.: Representation of the ECDF based binning algorithm. Represented is the two dimensional joint data space, therefore containing the realizations x, y of two random variables X, Y. The dots represent data points, i.e. combinations of realizations x, y. For each bin of X, an ECDF for the Y data in that bin is found and therefore the bin boundaries found; this process is recursively repeated. Each bin has index k for each dimension, i.e. k_x in x and k_y in y. These index vectors are then linearized into (i,j) form. Counting the transition from two bins (in these figure) will give the transition conditional probability $p(x_j, y_j | x_i, y_i)$. Figure taken from Brenner 2016 [6].

This approach applied to the current joint state space $\mathbf{\Phi}$ leads to [21]:

$$\mathbf{\Phi_k} = \begin{pmatrix} \Phi^{(1)} \in \left[\Phi^{(1)}_{k^{(1)}-1}, \Phi^{(1)}_{k^{(1)}}\right) \\ \cdots \\ \Phi^{(d)} \in \left[\Phi^{(d)}_{k^{(d)}-1}, \Phi^{(d)}_{k^{(d)}}\right) \\ \cdots \\ \Phi^{(n_{d,n})} \in \left[\Phi^{(n_{d,n})}_{k^{(n_{d,n})}-1}, \Phi^{(n_{d,n})}_{k^{(n_{d,n})}}\right) \end{pmatrix} \tag{3.10}$$

$n_{d,n}$: number of dimensions in the $\mathbf{\Phi}$ space
$k^{(1)}, ..., k^{(n_{d,n})}$: bin indices and $\mathbf{k} = (k^{(1)}, ..., k^{(n_{d,n})})$ bin index vector, $k^{(d)} = l + 1$ denote the 1D region where $x^{(d)} \in [x^{(d)}_l, x^{(d)}_{l+1})$ for $l \in [0, n_b - 1]$, $k^{(d)} = n_b$ for $x^{(d)} = x^{(d)}_{n_b+1}$

Its associated size vector is defined as [21]:

$$\mathbf{I_k} = \begin{pmatrix} \Phi^{(1)}_{k^{(1)}} - \Phi^{(1)}_{k^{(1)}-1} \\ \cdots \\ \Phi^{(d)}_{k^{(d)}} - \Phi^{(d)}_{k^{(d)}-1} \\ \cdots \\ \Phi^{(n_{d,n})}_{k^{(n_{d,n})}} - \Phi^{(n_{d,n})}_{k^{(n_{d,n})}-1} \end{pmatrix} \tag{3.11}$$

This size vector is used to construct the hypervolume of $\mathbf{\Phi}_k$, i.e. the multidimensional bins domain $\mathbf{\Omega}_k$ [21]:

$$\mathbf{\Omega_k} = (\mathbf{I}_k)_1 \times ... \times (\mathbf{I}_k)_{n_{d,n}}$$

The binning process for the joint $\mathbf{\Phi'}$ space is done in the same way, this leads to [21]:

$$\mathbf{\Phi'_k} = \begin{pmatrix} \Phi'^{(1)} \in \left[\Phi'^{(1)}_{k^{(1)}-1}, \Phi'^{(1)}_{k^{(1)}}\right) \\ \cdots \\ \Phi'^{(d)} \in \left[\Phi'^{(d)}_{k^{(d)}-1}, \Phi'^{(d)}_{k^{(d)}}\right) \\ \cdots \\ \Phi'^{(n_{d,c})} \in \left[\Phi'^{(n_{d,c})}_{k^{(n_{d,c})}-1}, \Phi'^{(n_{d,c})}_{k^{(n_{d,c})}}\right) \end{pmatrix} \tag{3.12}$$

$n_{d,c}$: number of dimensions in the $\mathbf{\Phi'}$ space
$k^{(1)}, ..., k^{(n_{d,c})}$: bin indexes

Its associated size vector is defined as [21]:

$$\mathbf{I'_k} = \begin{pmatrix} \Phi'^{(1)}_{k^{(1)}} - \Phi'^{(1)}_{k^{(1)}-1} \\ \cdots \\ \Phi'^{(d)}_{k^{(d)}} - \Phi'^{(d)}_{k^{(d)}-1} \\ \cdots \\ \Phi'^{(n_{d,c})}_{k^{(n_{d,c})}} - \Phi'^{(n_{d,c})}_{k^{(n_{d,c})}-1} \end{pmatrix} \tag{3.13}$$

the obtained domain is:

$$\mathbf{\Omega'_k} = (\mathbf{I'}_k)_1 \times ... \times (\mathbf{I'}_k)_{n_{d,c}}$$

At this point the data has been discretized into $k^{(1)}, ..., k^{(n_{d,c})}$ and $k^{(1)}, ..., k^{(n_{d,n})}$ indexed bins and a unique current and next state (bin) index is computed. For every bin, each containing the data of a possible future state from all the possible future states of the joint state space $\mathbf{\Phi}$, an index is assigned. This index is used for the representation in transition matrix and correspond the column index j [21]:

$$j = 1 + \sum_{d=1}^{n_{d,n}} (k^{(d)} - 1) \cdot n_b^{d-1} \quad ; \quad j \in \mathcal{Z} : 1 \leq i \leq n_b^{n_{d,n}}$$

For every bin, each containing the data of a current state from all the possible current states of the state space $\mathbf{\Phi'}$, the index i is assigned [21]:

$$i = 1 + \sum_{d=1}^{n_{d,c}} (k^{(d)} - 1) \cdot n_b^{d-1} \quad ; \quad i \in \mathcal{Z} : 1 \leq i \leq n_b^{n_{d,c}}$$

At this point the i,j indices of each multidimensional bin containing the data of a determined state have been found and thus it is possible to compute all the entries of the TMs $p(\mathbf{\Phi}|\mathbf{\Phi'})$ [21]:

$$\mathbf{T}_{ij} = \frac{\int_{\mathbf{\Omega}_j} \int_{\mathbf{\Omega}'_i} p(\mathbf{\Phi}, \mathbf{\Phi'}) d\mathbf{\Omega}' d\mathbf{\Omega}}{\int_{\mathbf{\Omega}'_i} \int_{\mathbf{\Omega}} p(\mathbf{\Phi}, \mathbf{\Phi'}) d\mathbf{\Omega} d\mathbf{\Omega}'} = p(\mathbf{\Phi}_j|\mathbf{\Phi}'_i) \tag{3.14}$$

This formulation is simply a generalization which describes the process of counting the number of state changes from bin i ($\mathbf{\Omega}'_i$) (containing data from Φ') to bin j ($\mathbf{\Omega}_j$) (containing data from Φ). The state transition frequency is then divided by the number by the total number of transition starting from bin i and going in all the other different bins, obtaining thus the conditional probability of the specific state change $i \to j$.

TM size

The size of the TMs is determined by the number of bins used to classify the data. In this framework $n_b = 20$ bins are used and have been shown to be enough to provide an acceptable accuracy in the data classification [21]. The TMs size also depends on the random variables considered; for the angle and step length TM both the current and next joint spaces have dimensions $n_{d_c} = n_{d_n} = 2$, while for the velocity TM the current state space has dimension $n_{d_c} = 3$ and the next state space has dimension $n_{d_n} = 1$. Therefore the TMs sizes obtained are $20^2 \times 20^2$ and $20^3 \times 20^1$ respectively [21]. A low number of effective segments, i.e. data points, combined with the fixed number of bins, may lead to an inaccurate transition PDF estimation and ultimately to inaccurate results in the stochastic simulation.

Example construction of TM $p(\theta, ln(\widetilde{l})|\theta', y')$:

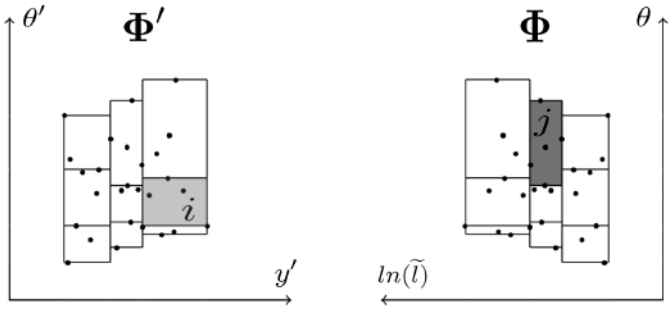


Figure 3.7.: Illustration of the two joint spaces, each discretized into $n_b = 9$ bins (in the used framwork $n_b = 20$). On the right the $i-$th bin is coloured in light grey while on the left the $j-$th bin is coloured in dark grey. The bin i contains the state data (y'_i, θ'_i) while the j bin contains the state data $(ln(\widetilde{l})_j, \theta_j)$. This figure is a modification of the one present in Brenner 2016 [6], and does not represent the real data, it is just a visualization (illustration) of a simplified case.

For the joint space $\mathbf{\Phi'}$, data points $(y', \theta') = (\Phi'^{(1)}, \Phi'^{(2)})$ with $n_{d,c} = 2$ are defined, therefore each bin has its own vectorial index $\mathbf{k} = (k^{y'}, k^{\theta'}) \equiv (k^{(1)}, k^{(2)})$ which is then linerized into a i index notation. For $\mathbf{\Phi}$, formed by the $(ln(\widetilde{l}), \theta) = (\Phi^{(1)}, \Phi^{(2)})$ data set , with $n_{d,n} = 2$; each bin has its own vectorial index $\mathbf{k} = (k^{ln(\widetilde{l})}, k^{\theta}) \equiv (k^{(1)}, k^{(2)})$ (not the same as above) which is then linerized into a j index notation. Each dimension is classified in $n_b = 20$ bins using the ECDF approach. The transitions frequency $i \rightarrow j$ leads to the definition of the TM element $p(\theta_j, ln(\widetilde{l})_j | \theta'_i, y'_i)$.

3.2.4. Next state selection

The TMs constructed with the MCS data are now utilized in the SDM simulation to produce new pathlines at low computational cost. The core of this process includes the selection of the new states j starting from the current state i. Having used a stencil table representation now comes handy since the new state selection is based on randomly iterating over all the j (with a specific i) and selecting a new index. This process has to consider the higher or lower probability of certain transitions (given by their frequency of occurring), therefore needs to follow the distribution of the transitions [6]. Inverse transform sampling meets this requirement and consist of generating random sample numbers from any PDF given its Cumulative Distribution Function (CDF) [32]. In this framework ECDF are used instead of CDF due to the data based nature of the task [6].

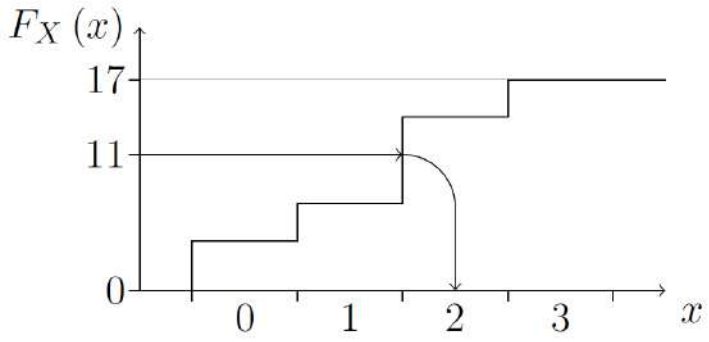


Figure 3.8.: Plot of the one dimensional ECDF $F_X(x)$ (unnormalized) as a function of its argument x. The x values are grouped into bins of width 1 (boundaries noticeable at the "steps" in the function). For this particular situation, $N_i = 17$ and the randomly choose k value is 11, which leads to the bin number 2 and therefore $j = 2$. Figure taken from Brenner 2016 [6].

The ECDF is created by "cumulative addition of all rows elements" [6], i.e. over all the $\mathbf{T}_{ij}$ for a fixed i. Following a similar procedure as eqn. (3.7), the next state index j is found by randomly selecting an ECDF $(F_X(x))$ value using an uniformly distributed integer $k \sim Unif[0, N_i - 1]$ $(k = F_X(x))$, with N_i the number of elements in the i-th row, and from the selected value find the function argument, i.e. the data value (x), which is the j-th bin [6]. In other words the

lowest index j so that $\mathbf{T}_{ij} > k$ is selected [20]; this can be easily done by "iteratively checking the matrix row elements until one of them is larger than the random value k, the corresponding column value is then found" [6]. It is also possible to work with a normalized distribution by dividing the ECDF by the number of elements (N_i) and using $k \sim Unif[0,1]$; both approaches leads to the same result [20].

3.2.5. Pathlines generation

Each particle at time $\widetilde{t} = 0$ is placed on its initial position $\widetilde{\mathbf{x}}_0 = (x(\widetilde{t} = 0), y(\widetilde{t} = 0))^T$ in the same manner as in MCS. The initial conditions for angle, effective step length and velocity are sampled from the MC pathlines initial data; i.e. $\theta, \widetilde{l}, \widetilde{u}$ are sampled from the joint distribution $(\theta_0, ln(\widetilde{l}_0))$ and from the PDF $p(ln(\widetilde{u}_0)|\theta_0, ln(\widetilde{l}_0))$ (generated following the same procedure as the other TMs) respectively [21]. The successive state is found by using the index iterating algorithm previously explained. The new tracer particle position and time at the n-th step is found with [21]:

$$\widetilde{\mathbf{x}}_{n+1} = \widetilde{\mathbf{x}}_n + \begin{bmatrix} cos\theta_n \\ sin\theta_n \end{bmatrix} \widetilde{l}_n \quad ; \quad \widetilde{t}_{n+1} = \widetilde{t}_n + \frac{\widetilde{l}_n}{\widetilde{u}_n} \tag{3.15}$$

Each new particle pathline is independently generated; all the pathlines form the so called "particle cloud".

3.2.6. SDM simulations settings

For each new tested medium characterization 25600 random walks were created, all using the same TMs (which are unique for each tested case). The pathlines are generated until the number of segments (RW steps) equals the minimum number of effective segments present in the reference MCS and the total time exceed 200. For the final plotted statistics, only the data up to the minimum particle exit time in MCS is used, which is not the same for each case.

3.3. Post-processing: statistics from MCS and SDM

The goal of all these simulation tools is estimating the particle cloud (plume) spreading and compare the results obtained from SDM to MCS to asses the efficiency and accuracy of the model.

To this end it is useful to consider the mean particle position and the particles position variance at each point in the domain. This information gives an insight about which point most particle are situated near and how far are the farthest particles and with which velocity the particle ensemble mean position is moving. The necessary data are the position and time of each particle $(\widetilde{\mathbf{x}}, \widetilde{t})$; using linear interpolation it is possible to get data along the steps of the MCS and SDM

simulations. The mean position in $x = x_1/l_Y$ and $y = x_2/l_Y$ are given as follows [20]:

$$\langle x(\widetilde{t})\rangle = \frac{1}{N_p}\sum_{p=1}^{N_p} x_p(\widetilde{t}) \quad ; \quad \langle y(\widetilde{t})\rangle = \frac{1}{N_p}\sum_{p=1}^{N_p} y_p(\widetilde{t}) \tag{3.16}$$

N_p : total number of particles
p : particles count

the position variance of the particles is given by [20]:

$$\sigma_x^2(\widetilde{t}) = \frac{1}{N_p}\sum_{p=1}^{N_p}(x_p(\widetilde{t}) - \langle x(\widetilde{t})\rangle)^2 \quad ; \quad \sigma_y^2(\widetilde{t}) = \frac{1}{N_p}\sum_{p=1}^{N_p}(y_p(\widetilde{t}) - \langle y(\widetilde{t})\rangle)^2 \tag{3.17}$$

The position data could now be also used to construct the position PDF $p_{\mathbf{X}}(\mathbf{x};t)$. Once $p_{\mathbf{X}}$ is known, it is possible to compute the concentration PDF $p_C(c;\mathbf{x},t)$, its mean value and variance.

3.4. Already tested medium characterizations

Monga *et al.* 2019 [21] have already tested and reported the results for the three different scenarios listed below:

- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 0°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 1600 field realizations, 80 pathlines tracked, $\sigma_Y^2 = 1$
- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 1600 field realizations, 12 pathlines tracked, $\sigma_Y^2 = 1$
- 45° tilted pressure gradient $h_2/h_1 = 1$, two pixel line fracture families: $\theta_{fr} = 0°$ and $\theta_{fr} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 1600 field realizations, 32 pathlines tracked, $\sigma_Y^2 = 1$

3.5. SDM limitations

The stochastic dispersion model is strongly dependant on the data it utilizes to simulate the dispersion, i.e. on the effective steps originated from the MC simulation. Therefore the main problems may lay in the definition of effective segments itself. Let's consider two cases of fractured porous fields having fixed and equal sized fracture families: one with low a fracture density and the other with high fracture density; this translates into a small number of fractures and a large number of fractures respectively. For the low density case, it is possible that the tracer

particle encounters and thus enters only a few fractures, for example two, leading to the definition of only 2-3 effective segments. While for the high density case, which represent a highly connected network, there is the possibility that a tracer particle entering a fracture, continues travelling inside fractures until the other domain boundary. This would also lead to a very small number of data points (effective segments). The same arguments can be made in the case of variable fractures size but fixed number of fractures and for the case of both parameters being variable until the desired density is reached. Therefore it make sense to think about a fracture density range in which the stochastic dispersion model is accurate and applicable.

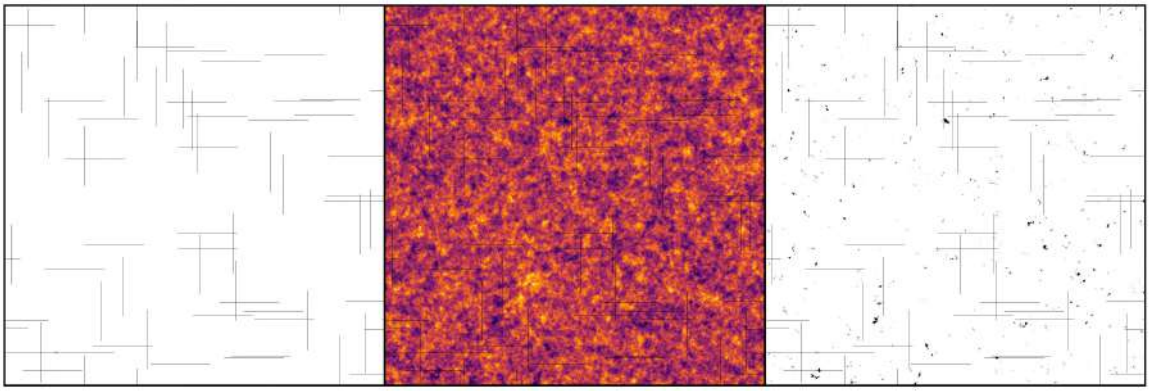


Figure 3.9.: Visual representation (illustration) of the highly conductive regions. On the left the possible effective data points locations, i.e. fractures exit points, used by the current framework. In the centre the reference crop-out image of a bigger random $ln(K)$ field realization with low fracture density; the color grading scheme used is the same as in Fig 3.1, with darker violet regions having high conductivity. On the right the proposed approach where also the high conductivity regions in the porous matrix are considered (represented as dark,irregularly shaped regions). The left and right pictures were obtained starting from the center one using image manipulation techniques in order to isolate the darker regions (more conductive) and show the concept.

Another possible problem could occur when considering very low conductivity fractures. This type of fractures act as obstacle for the fluid flow and particles movement. If a tracer particle does not enters fractures at all, then it is not possible to define effective segments. A possible solution to both problems could consist in defining effective segments between the exit points from all the regions with maximal conductivity or above a certain threshold. This definition still includes the fractures but also the small extremely conductive regions in the porous matrix. If a particle pass through this regions, then a data point is counted at the exit point. This approach should increase the total number of possible available effective data points, a visual representation of this concept can be found in Figure 3.9.

A limitation of the whole framework is the limited number of fractures types it is able to produce and the inability to use lengths and aperture distributions. Let's imagine two fields both with the same fracture density but one with all the fractures uniformly distributed in the central

region, while the other with the fractures uniformly distributed the top and bottom boundaries; even if the density is the same, the two resulting dispersion scenarios are completely different. The distance between fractures and their density in a certain region influence the particles path, therefore I think it would be useful to introduce new tunable parameters regarding the fractures, i.e. the fractures mean position and variance.

3.6. New tested medium characterizations

Considering the importance of the network connectivity for the transport process, the possible problems that the SDM may encounter and the already tested cases by Monga *et al* . 2019 [21]; the choice of new medium characterization consist of finding the lower and upper boundaries for the fracture density, and also look at the behaviour for different porous matrix heterogeneities (change the variance).

The choice of tested fracture densities is dictated by the minimum number of effective segments that can be originated, i.e. the number of effective data points used to construct the TMs. Also a similar case (missing one) to the ones treated in Monga *et al.* 2019 [21] is analysed to check the overall accuracy of the model. For the varied log-conductivity variance σ_Y^2 cases the fracture parameters are the same as the base cases investigated in Monga *et al.* 2019 [21].

For low fracture densities the case with longitudinal fractures might be critical ($\theta_{fr} = 0$) due to the higher chance of a particle only entering few fractures and travelling most of the domain length through them; obtaining therefore too few effective data points. For high fracture densities the case with two fracture families is likely to be critical due to the higher connectivity which might lead to particles never exiting the fractures.

Tests on smaller domains were conducted in order to decide which fracture density to choose. Fields with fracture density $f = 1\%$ and $f = 2\%$ would lead to the definition of only 2 effective segment in the case of longitudinal fractures, which would not be enough to obtain accurate TMs estimations. Therefore 3% was chosen as being a middle ground between the already known 5% cases and the inaccurate $1,2\%$ cases.

From the definition of effective segments, it is easy to see that the number of effective segments as a function of the fracture density, should initially increase and then decrease after reaching its maximum (bell shaped curve). This hypothesis was proven right with the downscaled domains tests for all possible fracture families combinations. The originated curves in Fig 3.10 have a maximum value slightly shifted towards the low densities and for the cases of $\theta_{fr} = 0°$ and $\theta_{fr} = 0°, 90°$ the maximum value was a lot smaller compared to the one obtained in the $\theta_{fr} = 90°$ test case (tested but not in the plot). This maximum difference shows that in fields with fractures perpendicular to the mean pressure gradient, more smaller effective segments are originated; showing how the tracer particles tend to travel in the pressure gradient direction even after entering high conductivity fractures perpendicular to it, i.e. particles exit immediately the transverse fractures if no connection with a longitudinal fracture is present. The accuracy of the

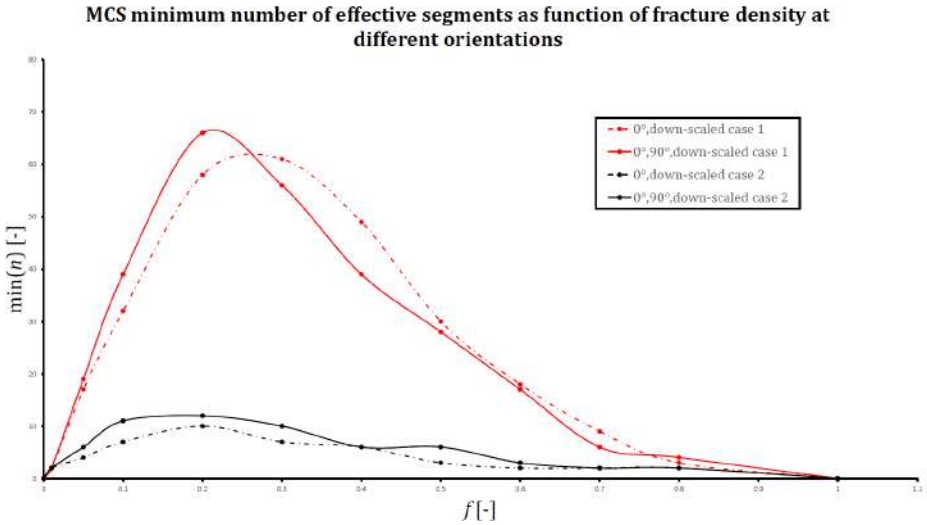


Figure 3.10.: Minimum number of effective segments n produced by MCS plotted as function of fracture density. Down-scaled fields characteristics: size 128×64, resolution 512×256 orthogonal cells, variance $\sigma_Y^2 = 1$, correlation length $l_Y = 4$ cells in both directions. Case 1: $l_{fr} = 0.75$, $a_{fr} = 0.25$. Case 2: $l_{fr} = 10$, $a_{fr} = 0.25$. For case 1 the fracture size was chosen so that the number of fracture present at $f = 5\%$ would be similar to the number of fractures present in the full scale simulation at the same density. For case 2 the fracture size was randomly chosen bigger than case 1 in order to see if the number of fractures had an impact on the curves shapes.

model, considering only statistical errors, should follow the behaviour seen in Fig 3.10, because more effective segments produce more data, leading to a better statistical representation of the original pathlines. Therefore it should be possible to make predictions about the model limitations. From Fig 3.10 it is noticeable that different fractures geometries have similar behaviours, therefore this behaviour is neither linked to the number of fractures in the domain nor to their geometry, therefore the prediction should be applicable also to the full scale simulation.

Due to the high computational times required for high f values, $f = 20\%$ was selected to be tested at full scale since it would also provide informations regarding the approximate highest reachable accuracy. Assuming that only statistical errors are present, the upper density boundary is predicted to be between $f = 70\%$ and $f = 80\%$ as shown in Fig. 3.10. The orientation of the pressure gradient was kept longitudinal (no inclination angle) for the 3% in order to have most tracer particles travel along the fractures. For density 20% the inclination angle of the pressure gradient was also kept null so that this case could be compared to the 5% also under consideration that the connectivity would not change much compared to an inclined pressure gradient. The choice of variances was based on the extreme ones used to validate the PMVP model by Meyer *et al.* 2013 [19]. For these cases the pressure gradient was also kept horizontal in order to be able to compare them to the one present in Monga *et al.* 2019 [21]. The new tested cases are subdivided in two categories:

Fracture density:

- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 0°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 3\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 80 pathlines tracked, $\sigma^2_Y = 1$
- longitudinal pressure gradient $h_2/h_1 = 0$, two pixel line fracture families: $\theta_{fr} = 0°$ and $\theta_{fr} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 32 pathlines tracked, $\sigma^2_Y = 1$
- longitudinal pressure gradient $h_2/h_1 = 0$, two pixel line fracture families: $\theta_{fr} = 0°$ and $\theta_{frac} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 20\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 32 pathlines tracked, $\sigma^2_Y = 1$

Field variance:

- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 0°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 80 pathlines tracked, $\sigma^2_Y = 4$
- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 12 pathlines tracked, $\sigma^2_Y = 4$
- longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 90°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 12 pathlines tracked, $\sigma^2_Y = 1/16$
- (*) longitudinal pressure gradient $h_2/h_1 = 0$, one pixel line fracture family: $\theta_{fr} = 0°$, fracture length $l_{fr} = 20$, fracture aperture $a_{fr} = 0.0625$, fracture density $f = 5\%$, fracture conductivity weight $Y_{fr} = 4$. MCS: 800 field realizations, 80 pathlines tracked, $\sigma^2_Y = 1/16$.
  (*) Due to problems to the computing infrastructure and the stochastic simulation itself, partial data is present for this case, the results and problems concerning this case are reported in Appendix C.

3.7. IFD framework summarized

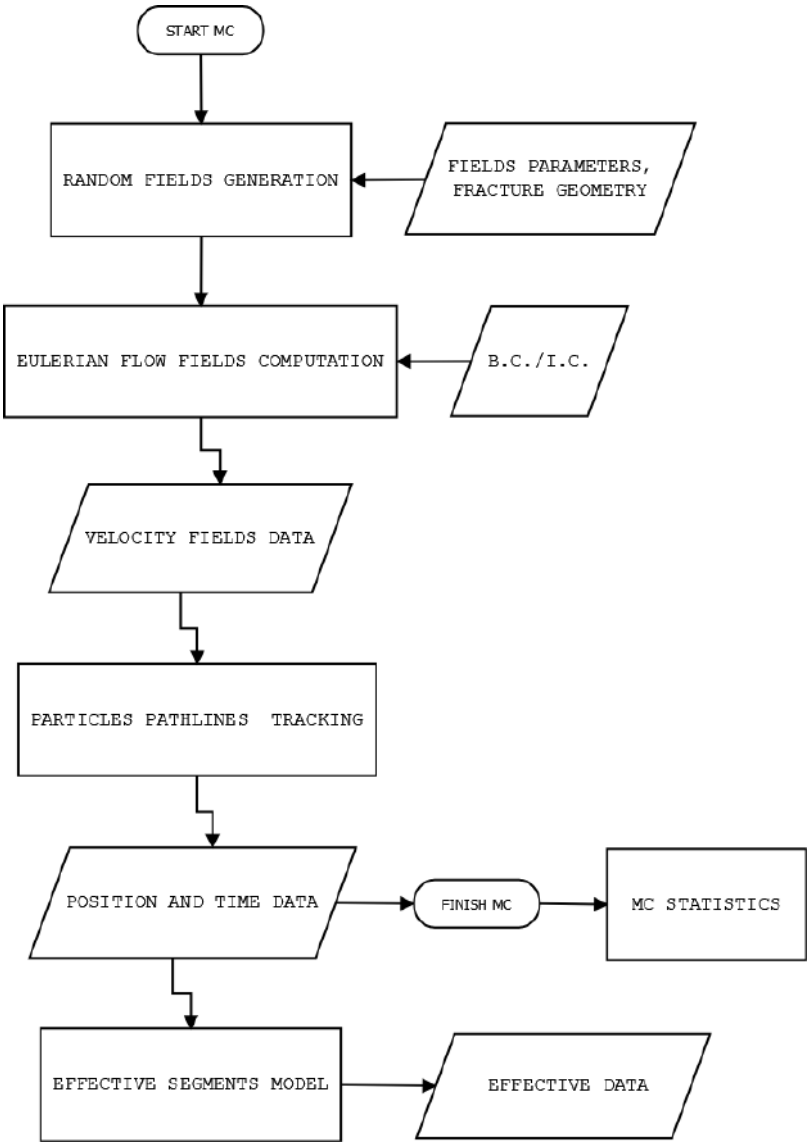


Figure 3.11.: Flowchart of the data gathering process with means of MCS

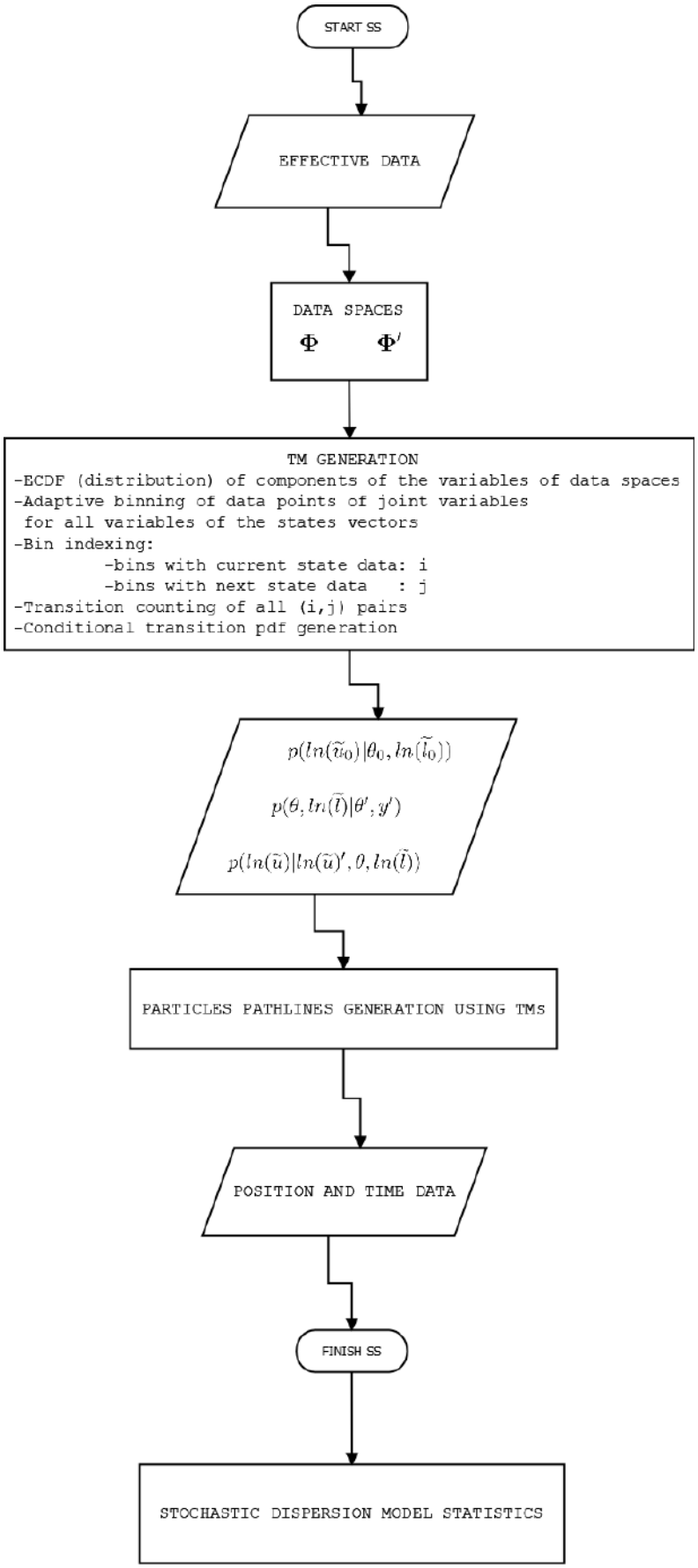
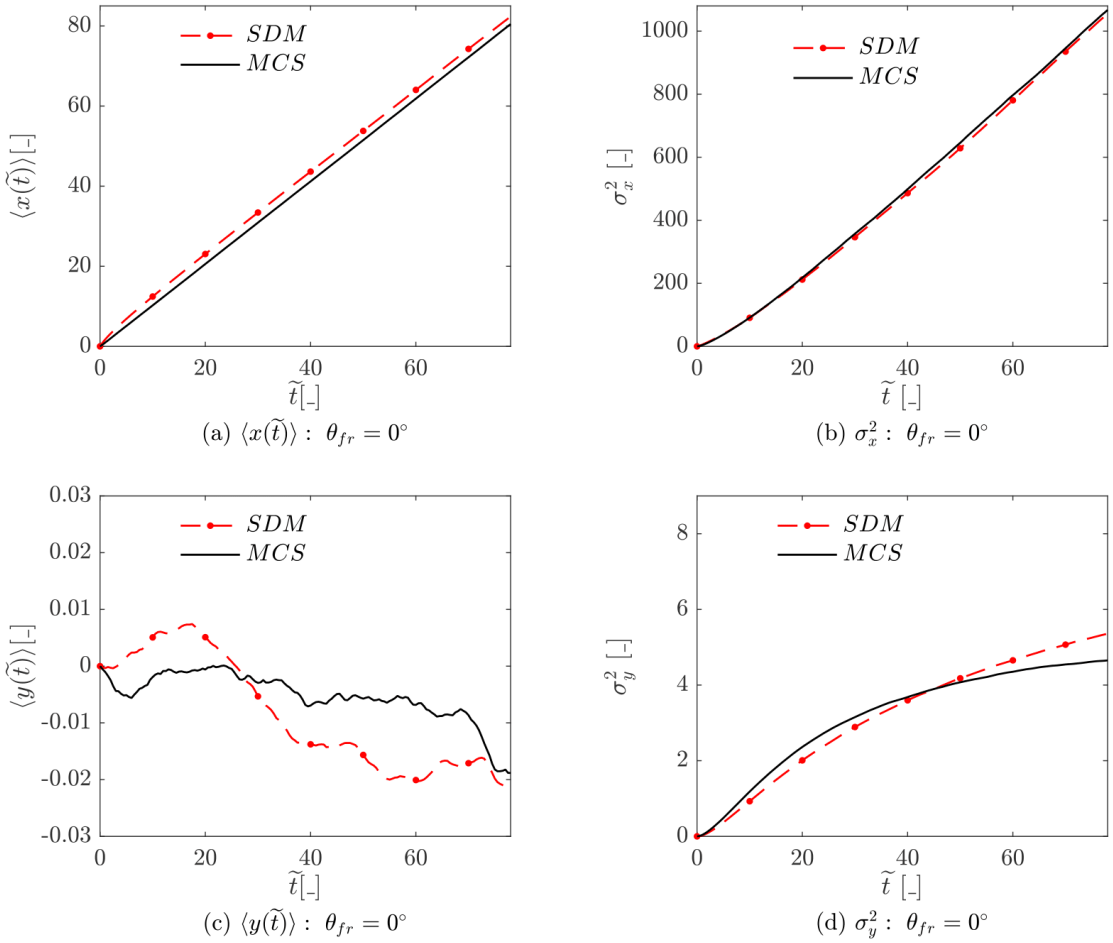


Figure 3.12.: Flowchart of the simulation process with means of the SDM

4. Results



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (b) σ_x^2 : $\theta_{fr} = 0°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (d) σ_y^2 : $\theta_{fr} = 0°$

Figure 4.1.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 3% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 78$.

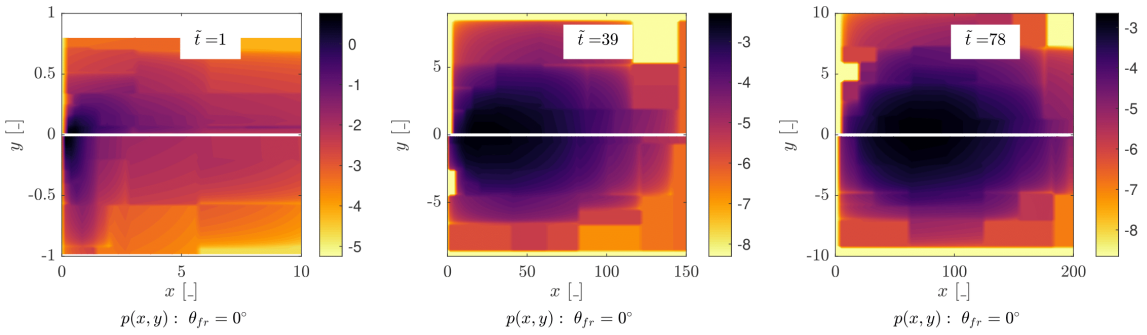


Figure 4.2.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 3% with log-hydraulic conductivity variance $\sigma^2_Y = 1$. Logarithmic colorbar scaling.

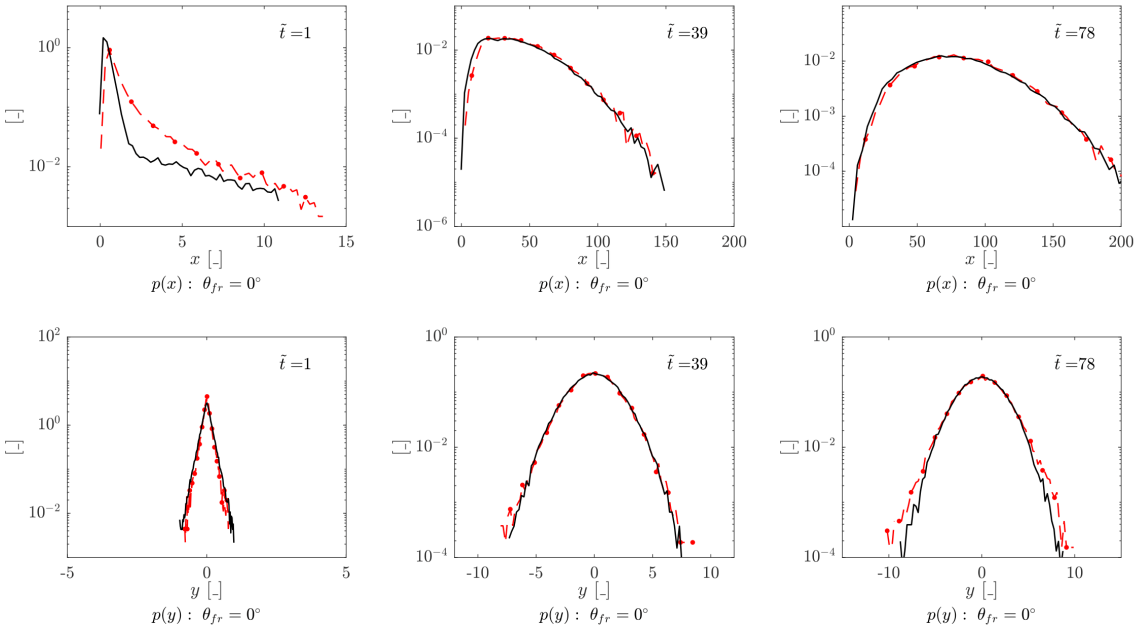
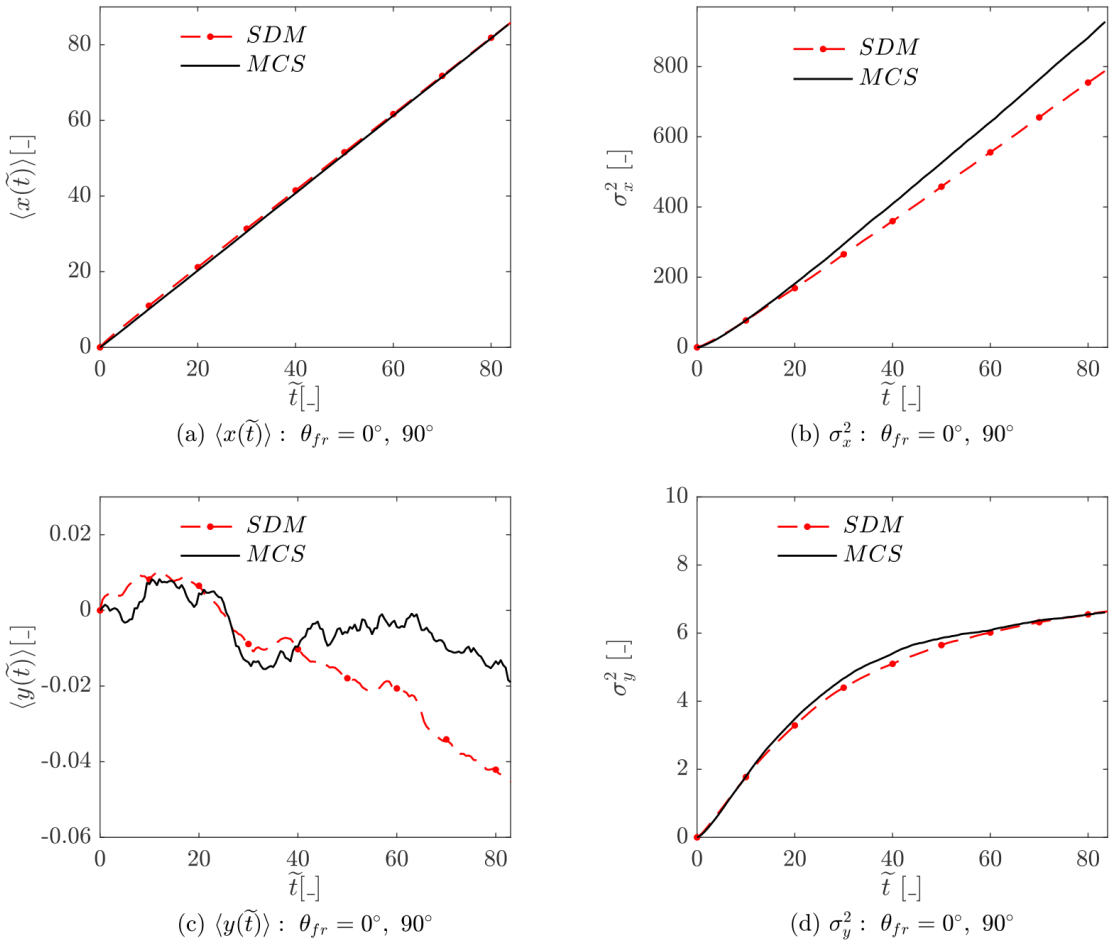


Figure 4.3.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 3% with log-hydraulic conductivity variance $\sigma^2_Y = 1$.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 0°,\ 90°$

(b) σ_x^2 : $\theta_{fr} = 0°,\ 90°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 0°,\ 90°$

(d) σ_y^2 : $\theta_{fr} = 0°,\ 90°$

Figure 4.4.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 83$.

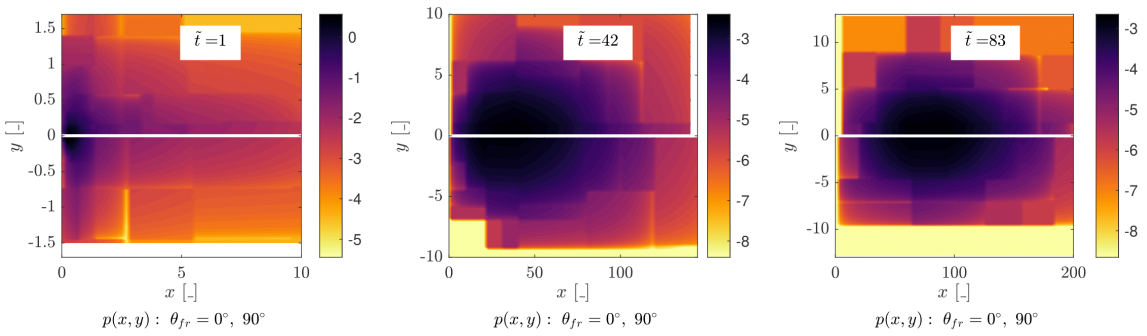


Figure 4.5.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$. Logarithmic colorbar scaling.

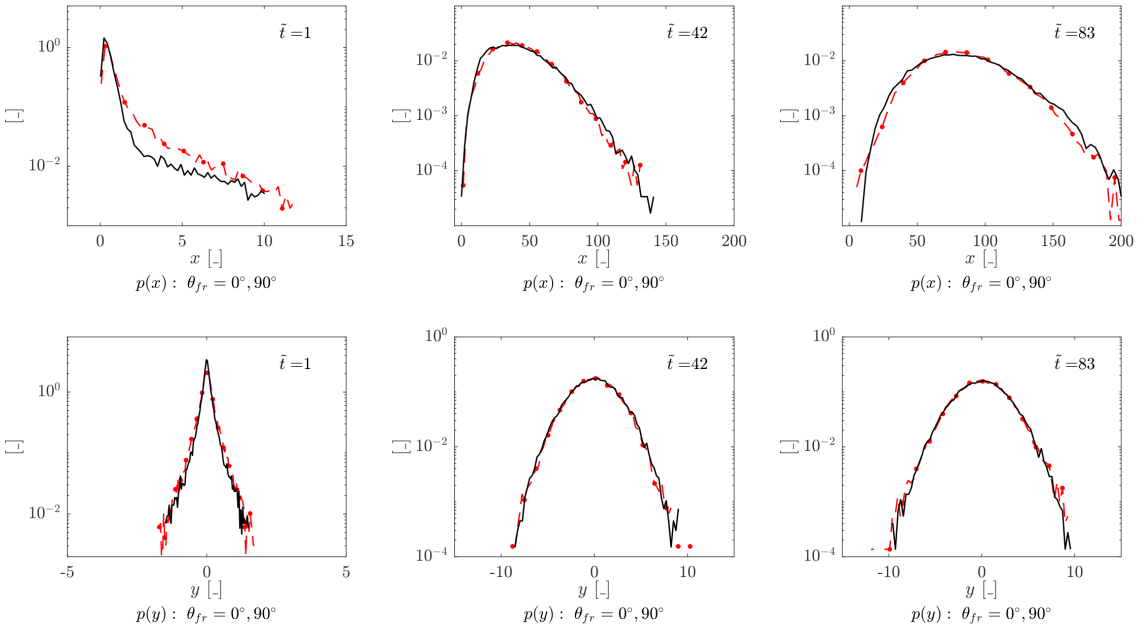
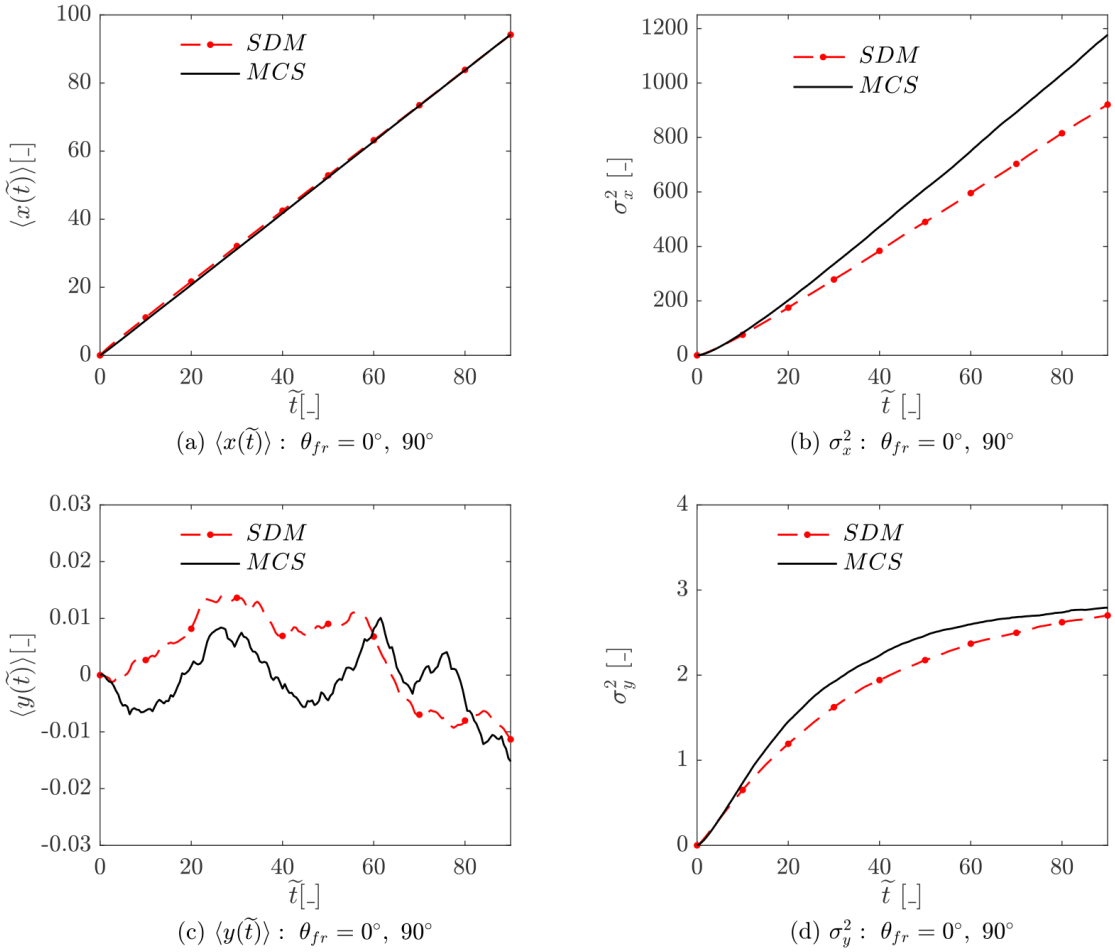


Figure 4.6.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 0°,\ 90°$ (b) σ_x^2 : $\theta_{fr} = 0°,\ 90°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 0°,\ 90°$ (d) σ_y^2 : $\theta_{fr} = 0°,\ 90°$

Figure 4.7.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 20% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 90$.

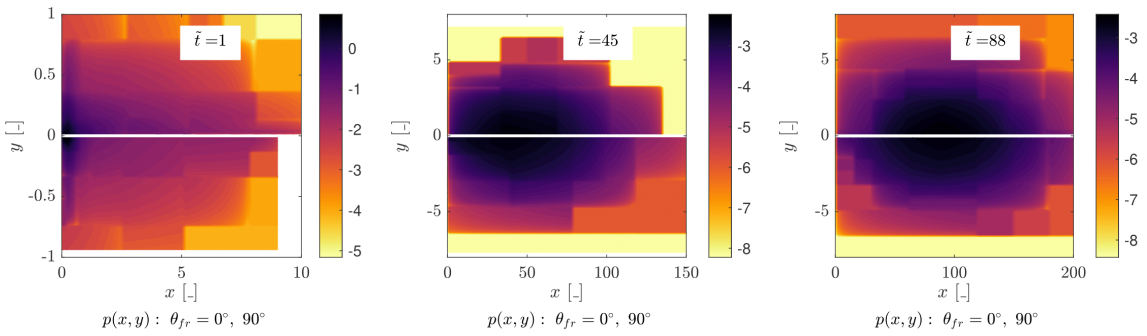


Figure 4.8.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 20% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$. Logarithmic colorbar scaling.

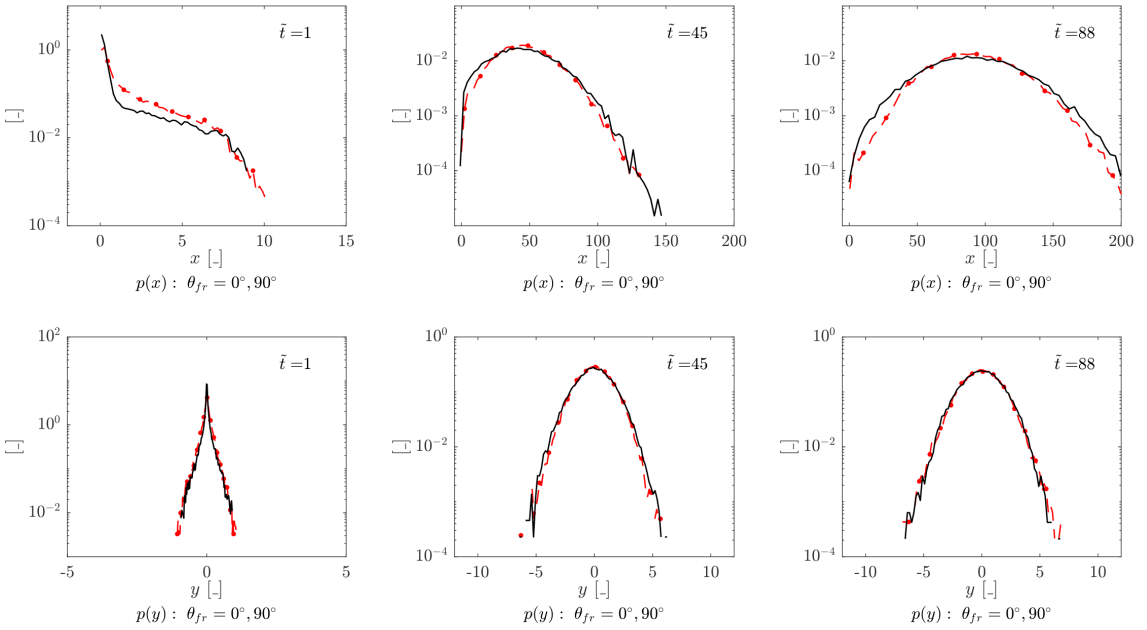
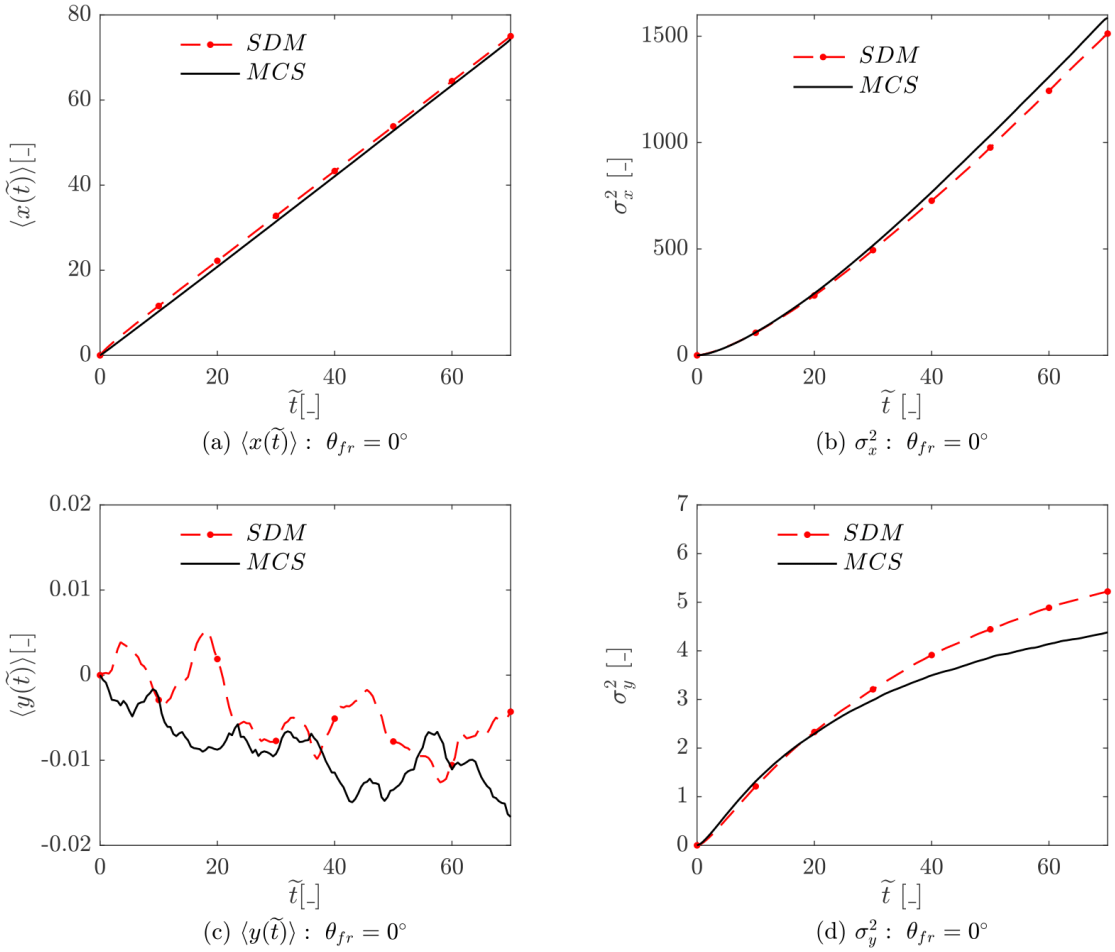


Figure 4.9.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 20% with log-hydraulic conductivity variance $\sigma_Y^2 = 1$.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (b) σ_x^2 : $\theta_{fr} = 0°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (d) σ_y^2 : $\theta_{fr} = 0°$

Figure 4.10.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 70$.

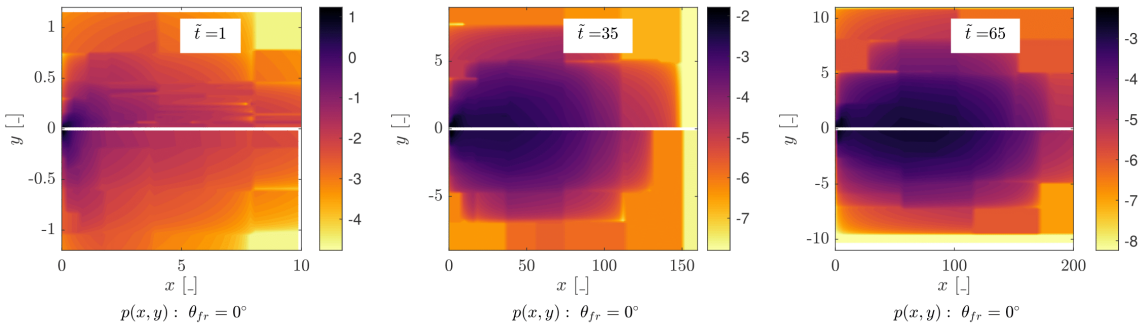


Figure 4.11.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$. Logarithmic colorbar scaling.

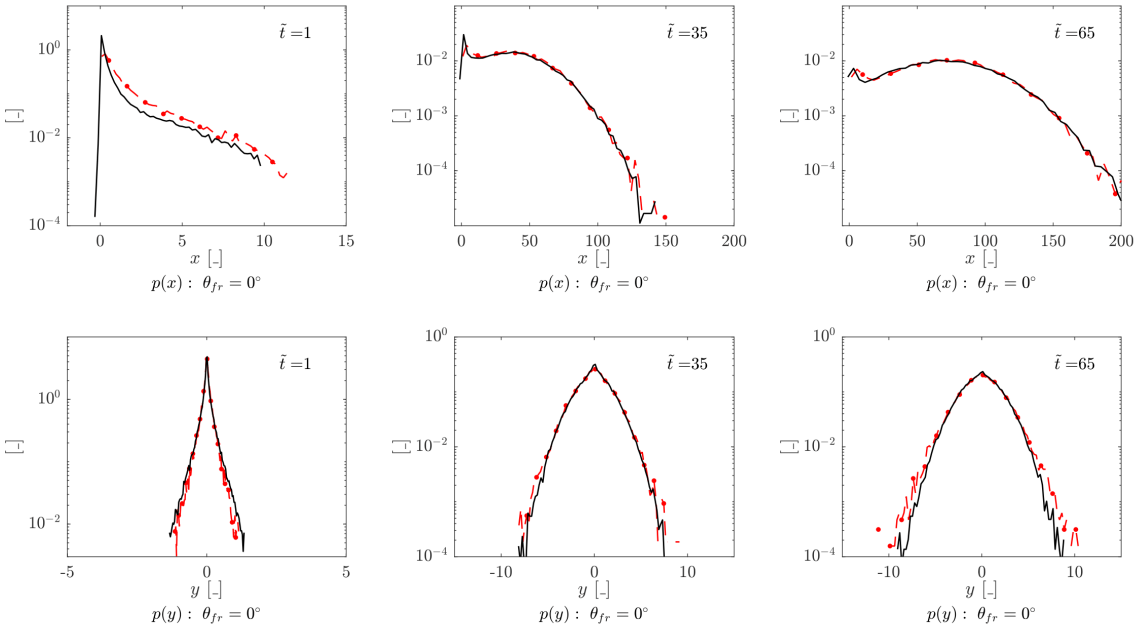
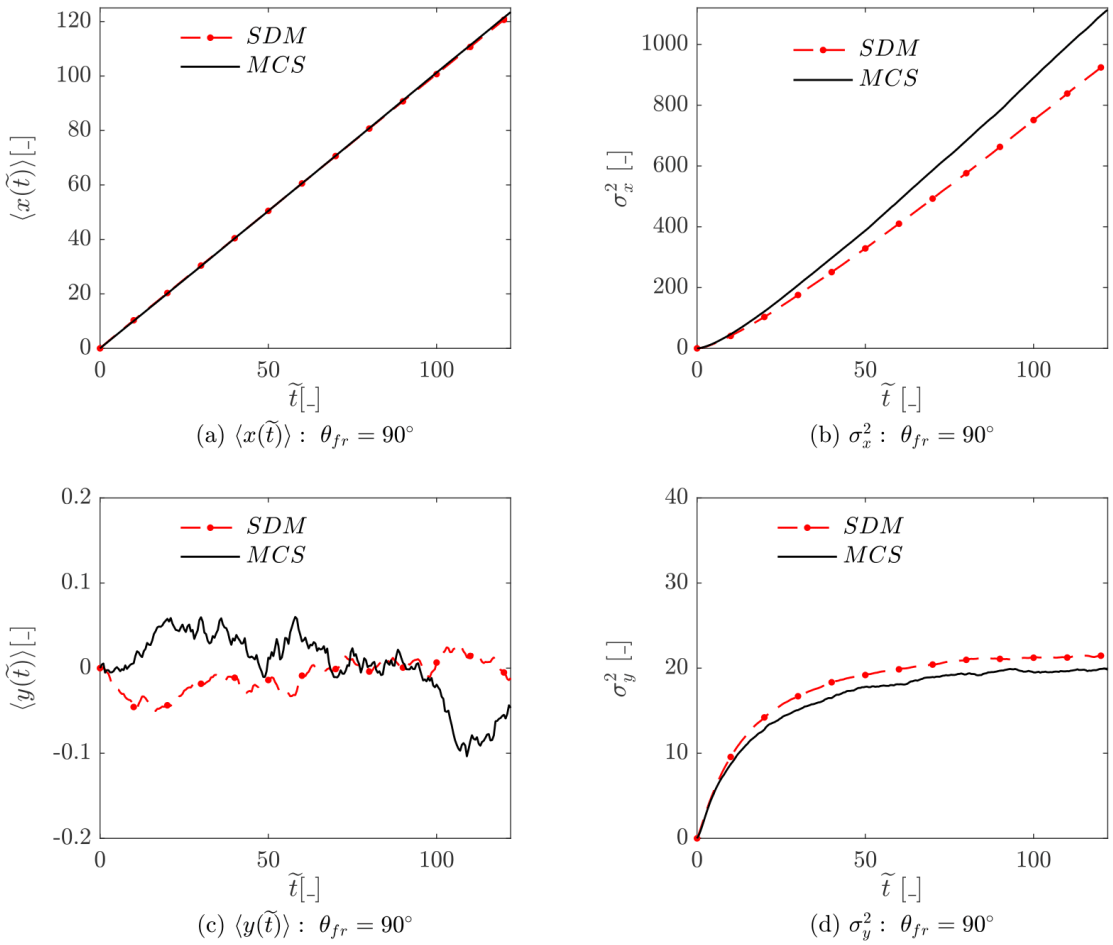


Figure 4.12.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 90°$ (b) σ_x^2 : $\theta_{fr} = 90°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 90°$ (d) σ_y^2 : $\theta_{fr} = 90°$

Figure 4.13.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 122$.

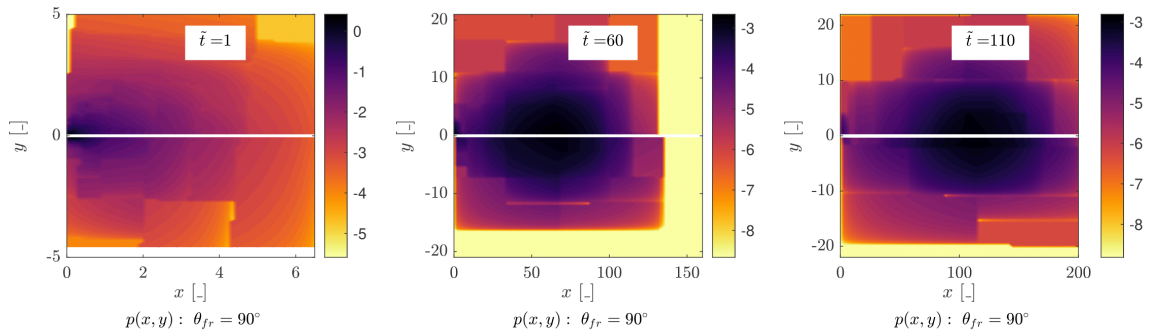


Figure 4.14.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$. Logarithmic colorbar scaling.

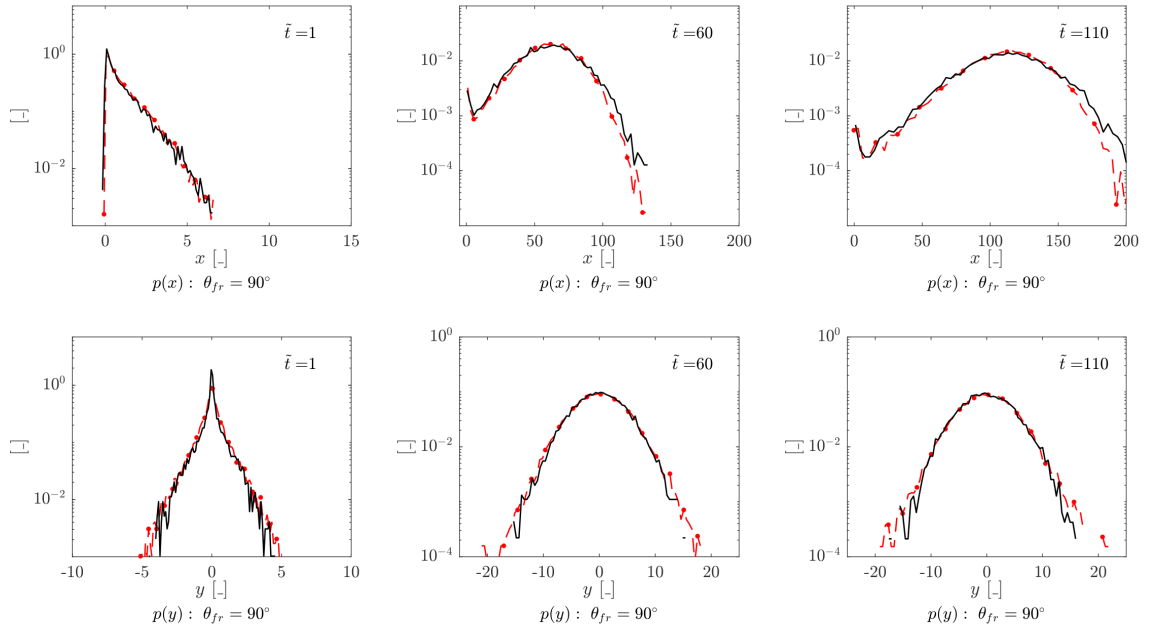
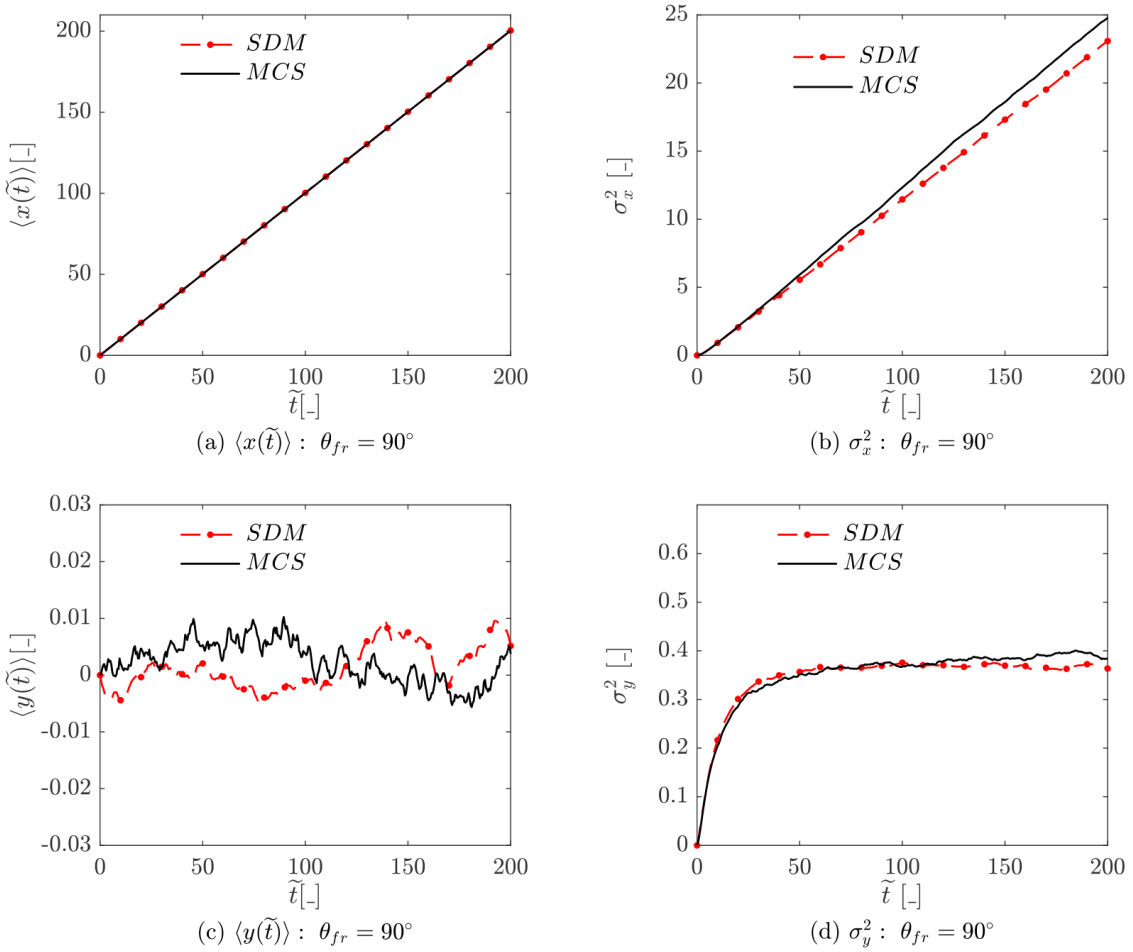


Figure 4.15.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 4$.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 90°$ (b) σ_x^2 : $\theta_{fr} = 90°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 90°$ (d) σ_y^2 : $\theta_{fr} = 90°$

Figure 4.16.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1/16$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 200$.

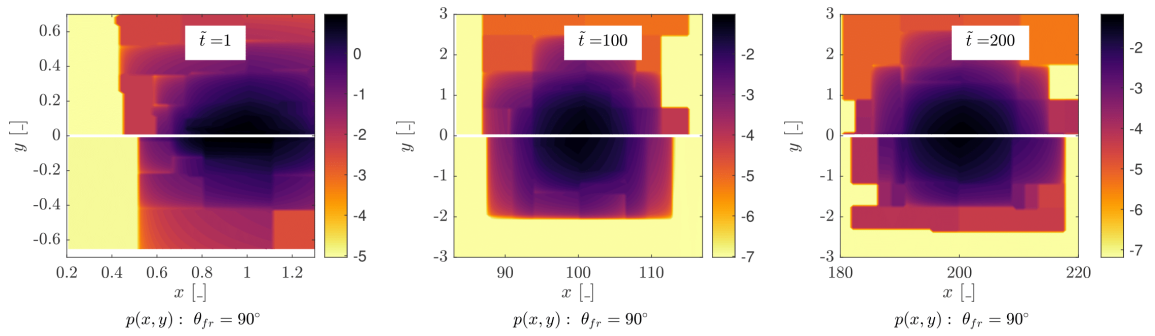


Figure 4.17.: Joint position PDF at different times with SDM data (upper-half) and with reference MCS data (lower-half). The joint PDFs are obtained using a Density Estimation Tree (DET) algorithm [21]. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1/16$. Logarithmic colorbar scaling.

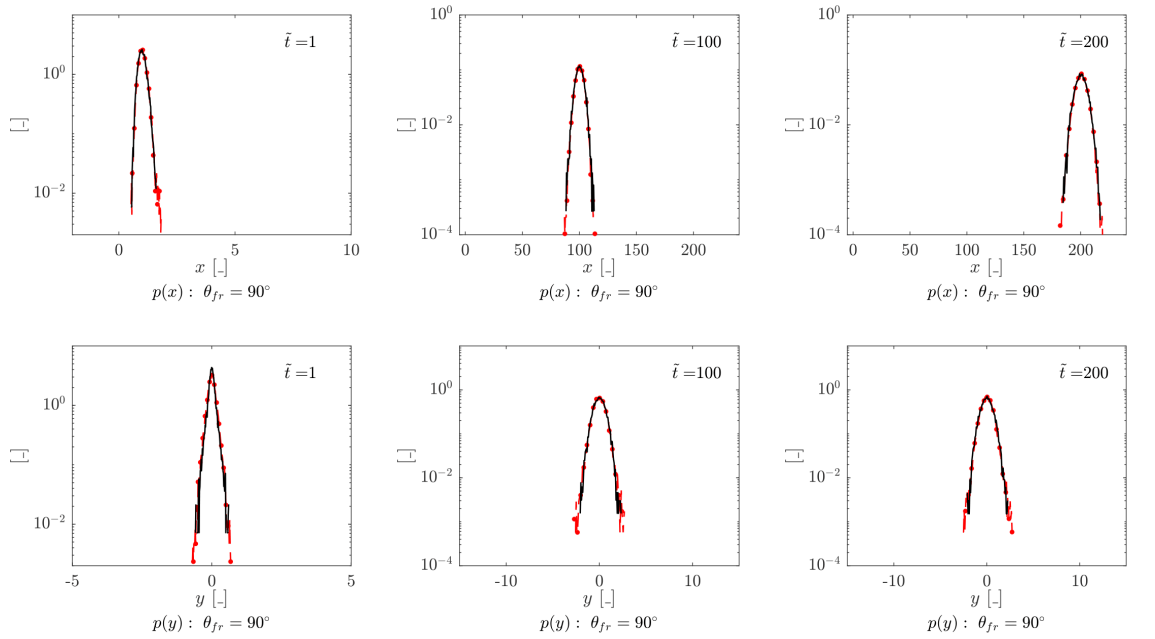


Figure 4.18.: Marginal position PDF at different times with SDM data (dashed red line) and reference MCS data (full black line). Upper row for longitudinal position, lower row for transverse position. Logarithmic scaling for $p(\bullet)$. Field fracture density 5% with log-hydraulic conductivity variance $\sigma_Y^2 = 1/16$.

5. Analysis

5.1. Longitudinal mean $\langle x(\widetilde{t})\rangle$

For all the tested cases there is an excellent agreement between the SDM longitudinal mean position as a function of time curve and the MCS one. Overall the Stochastic Dispersion Model predicts the longitudinal mean position of the particle cloud very well.

5.2. Transverse mean $\langle y(\widetilde{t})\rangle$

Each case has a different behaviour for the transverse mean. Overall all curves stay bounded in very small intervals with respect to the domain width, and the SDM is able to predict accurately the cloud transverse mean position.

5.3. Longitudinal σ_x^2 and transverse variance σ_y^2

5.3.1. Fracture density case: $\sigma_Y^2 = 1$, $f = 3\%$

For fracture density 3% with longitudinal fractures, there is a very good agreement between the SDM and MCS predicted cloud position longitudinal variance. Before $\widetilde{t} = 40$, the SDM transverse variance (Fig 4.1 (d)) is slightly smaller than the MCS one, but at late times, towards the asymptotic behaviour of MCS (not reached yet), the transverse variance is over estimated by the SDM.

5.3.2. Fracture density case: $\sigma_Y^2 = 1$, $f = 5\%$

For higher densities, 5%, with both longitudinal and transverse fractures, the SDM under estimates the longitudinal position variance , but the transverse variance is very well predicted as can be seen in Fig 4.4 (d). The cause of the improvement compared to the 3% density is not know since statistical errors and/or bias in the used pathline variables could be present. Also the maximum reached transverse variance value is close between the two cases.

5.3.3. Fracture density case: $\sigma_Y^2 = 1$, $f = 20\%$

For the 20% density case, with same fracture families as the 5% case, the longitudinal variance is still under estimated, and this time the difference between SDM and MCS is grater.

The transverse variance is in close agreement between the two curvers for very early and late times. Towards late times the curves are approaching an asymptotic behaviour, i.e. the particle plume is not spreading further in y direction.

5.3.4. Field variance case: $\sigma_Y^2 = 4$, $f = 5\%$:

Looking at Fig 4.10 (b) and 4.13 (b) it can be seen that for fields containing longitudinal fractures, the longitudinal position variance is well predicted by SDM compared to fields containing transverse fractures, which have a worse performance of the model. On the other hand the transverse position variance is over estimated by the SDM for the first case and in close agreement between SDM and MCS in the second. For the second case the transverse variance is able to reach an asymptotic limit.

5.3.5. Field variance case: $\sigma_Y^2 = 1/16$, $f = 5\%$:

For this low field variance case the SDM is able to predict very well the transverse variance and also fairly well, compared to other cases, the longitudinal variance. Being a field with fractures perpendicular to the mean pressure gradient, the surrounding porous matrix has a bigger impact on the solute spreading. Since in this case the matrix is not very heterogeneous due to the low conductivity variance, also the matrix does not have much impact on the solute spreading; therefore the particle cloud does not spread extensively neither in longitudinal, nor in transverse direction. This behaviour is very noticeable in Fig. 4.17, where the particle cloud has sharp boundaries and a standard deviation in longitudinal direction of only 20, while in other cases the cloud spread for the whole 200 units distance. Looking at the marginal position PDF plots of Fig 4.18, it is clear that the position variances are very small, leading to a more "compact dispersion"; this characteristic is also readable from the low maximum value of the position variances in Fig. 4.16 (b) and (d).

5.4. Longitudinal $p(x(\widetilde{t}))$ and transverse $p(y(\widetilde{t}))$ position marginal PDF

In all the tested scenarios the SDM (red dashed line) predicts very well the marginal position PDF when compared to the reference MCS curve (full black line). For the low variance $\sigma_Y^2 = 1/16$ case and all the cases transverse position PDF from mid to late times, the particle cloud shape tends toward a Gaussian form. Apart from these instances, the cloud marginal PDF has always a non-Gaussian shape [21] as expected for dispersion in very heterogeneous fractured field.

5.5. Fracture density effect on dispersion

From the fracture density cases it is shown that increasing the fracture density f leads to an increase in the rate at which the variance in x direction increase; this is translated into a faster

dispersion in x direction. For density 5%, the variance in transverse direction reaches higher values than in the 20% case. Both behaviours can be attributed to the fact that in the higher density case both the number of longitudinal fractures and transverse fractures has increased, but their effect on particles transport is not the same. Fractures aligned with the mean pressure gradient act as long fast path in the gradient direction, while transverse fracture only have their small width aligned with the pressure gradient, therefore particles tend to travel longer in longitudinal fractures and exit immediately transverse fractures. The increase in the number of longitudinal fractures has a greater impact on transport than the increase in the number of transverse fractures, resulting in less transverse spreading and faster longitudinal spreading for high densities. For both cases the travel times do not differ much between each other; therefore changing fracture density has much more impact on the plume spreading rather than the travel time.

5.6. Fracture density effect on model accuracy

The fracture density has a direct impact on the number of effective segments generated. More effective segments should lead to a more precise final result since more states are considered and therefore the dispersion is statistically better described (possibly less statistical errors). Looking at the difference between the position variances σ_x^2, σ_y^2 obtained with MCS and SDM in Fig 4.4 and Fig 4.7, it is noticeable that for the higher density field ($f = 20\%$, lowest number of effective segments 47) the SDM perform worse than in the lower density ($f = 5\%$, lowest number of effective segments 38) field case. Looking at the absolute normalized difference of the curves data at mid and late times (approximative results):

$$\Delta_{x,f} = \Big|\frac{\Delta\sigma_x^2}{\sigma_{x,MCS}^2}\Big|_{f=20\%} - \Big|\frac{\Delta\sigma_x^2}{\sigma_{x,MCS}^2}\Big|_{f=5\%} \approx \begin{cases} 7.7\% & ; \quad \widetilde{t} = 40 \\ 6.8\% & ; \quad \widetilde{t} = 80 \end{cases}$$

$$\Delta_{y,f} = \Big|\frac{\Delta\sigma_y^2}{\sigma_{y,MCS}^2}\Big|_{f=20\%} - \Big|\frac{\Delta\sigma_y^2}{\sigma_{y,MCS}^2}\Big|_{f=5\%} \approx \begin{cases} 7.0\% & ; \quad \widetilde{t} = 40 \\ 4.0\% & ; \quad \widetilde{t} = 80 \end{cases}$$

the SDM model is consistently performing slightly worse in the higher density case at mid to late travel times. Therefore it seems that there is some characteristic of higher density fields that has more impact in the accuracy than the number of effective data points, which the SDM model is not considering. It is also possible that this difference is caused by a too low binning resolution at high density, i.e. the $n_b = 20$ bins per dimensions were not able to capture well enough the spectrum of states. Prediction regarding the model maximal reachable density are not possible (at the moment) due to the still unknown cause of the accuracy worsening discussed above; bias in the used pathline variables or statistical errors could be de cause.

5.7. Log-conductivity field variance effect on dispersion

Considering the data obtained by Monga *et al.* 2019 [21] ($\sigma_Y^2 = 1$ cases), and the data of this thesis, it is possible to compare the fields with $f = 5\%$ and $\theta_{fr} = 0°$ or $\theta_{fr} = 90°$, for different

field variance values $\sigma_Y^2 = (\frac{1}{16}, 1, 4)$. For fields having transverse fractures, the travel time (i.e. first particle exit time) decreases as the variance increase.

$$[\sigma_Y^2 = \frac{1}{16},\ \theta_{fr} = 90^\circ :\ \widetilde{t} = 200]\ ;\ [\sigma_Y^2 = 1,\ \theta_{fr} = 90^\circ :\ \widetilde{t} = 175]\ ;\ [\sigma_Y^2 = 4,\ \theta_{fr} = 90^\circ :\ \widetilde{t} = 122]$$

This can be explained by the increased presence of high conductivity regions in the porous matrix for high σ_Y^2 values, these regions provide faster pathways than the transverse fractures (since they are not aligned with mean pressure gradient), resulting in lower travel times. Even if the travel times become smaller, the particle cloud is still able to spread in all directions; with increasing conductivity variance the position variance increases in both directions. In general for fields with transverse fractures (perpendicular to the mean pressure gradient) the surrounding matrix has more impact on the solute spreading; therefore if the porous matrix is quite homogeneous (low conductivity variance) the particle cloud will not spread any further in longitudinal direction with time due to the absence of high conductivity regions in the matrix. If the variance is higher, high conductivity regions will be present and the spreading in both longitudinal and transverse direction will be facilitated. This behaviours can be easily seen in Figure 4.14 and 4.17, where in the latter the cloud behaves more like a travelling particles packet with very little spreading in longitudinal direction.

The travel times in presence of only longitudinal fractures do not vary much when changing σ_Y^2, this is due to the higher impact that fractures aligned with the mean pressure gradient have.

$$[\sigma_Y^2 = 1,\ \theta_{fr} = 0^\circ :\ \widetilde{t} = 70]\ \ ;\ \ [\sigma_Y^2 = 4,\ \theta_{fr} = 0^\circ :\ \widetilde{t} = 70]$$

In these cases the tracer particles travel mostly in the (horizontal) fractures, therefore the changes applied to the surrounding field does not have much impact. The position variance in both direction increases (slightly) when increasing the field variance.

5.8. Log-conductivity field variance effect on model accuracy

As for the fractures cases, also the field variance seems to influence the model accuracy when increased (Fig. 4.13, 4.16). Looking at the absolute normalized difference of the curves data at mid times for the fields (variance cases) having transverse fractures, the following results are obtained (approximative results):

$$\Delta_{x,\sigma_Y^2} = \Big|\frac{\Delta\sigma_x^2}{\sigma_{x,MCS}^2}\Big|_{\sigma_Y^2=4} - \Big|\frac{\Delta\sigma_x^2}{\sigma_{x,MCS}^2}\Big|_{\sigma_Y^2=1/16} \approx \begin{cases} 8.5\% & ;\ \widetilde{t} = 100 \\ 7.6\% & ;\ \widetilde{t} = 120 \end{cases}$$

$$\Delta_{y,\sigma_Y^2} = \Big|\frac{\Delta\sigma_y^2}{\sigma_{y,MCS}^2}\Big|_{\sigma_Y^2=4} - \Big|\frac{\Delta\sigma_y^2}{\sigma_{y,MCS}^2}\Big|_{\sigma_Y^2=1/16} \approx \begin{cases} 6.0\% & ;\ \widetilde{t} = 100 \\ 6.0\% & ;\ \widetilde{t} = 120 \end{cases}$$

The SDM is performing worse at high field variances too. Increasing the field log-conductivity variance leads to more complex particle paths. Therefore the decrease in accuracy might be linked to the effective segments kinematic variables not being very representative of the original

high resolution path at high σ_Y^2 (e.g. inaccurate averaged velocity). On the other hand, in fields with transverse fractures, the resulting effective segments have generally short lengths, and therefore, the previously proposed cause seems to fall short. In fields with transverse fractures, the correlations are prolonged. This prolongation increases for high σ_Y^2, possibly leading to higher errors.

5.9. Asymptotic transverse dispersion

This behaviour (Fig. 4.16 (d), 4.13 (d), 4.7 (d)) is caused by pathlines not intersecting with each other [21]. The pathlines at the extremes in transverse direction thus act as bounds for all the pathlines contained between them, leading to a maximal reachable transverse spreading (asymptotic) defined by these extreme pathlines. The condition of no intersection derives from the absence of mixing (no PSD for $Pe \gg 1$) in the considered framework [21].

5.10. SDM improvements

In the code the creation of multi-Gaussian fields with pixel line fractures whose lengths (x) follows exponential, log-normal and gamma distributions has been implemented. For exponentially and log-normally distributed fractures the input parameter l_{fr} of each fracture family is treated as the mean of the family specific lengths distribution. Random samples are then extracted from these distributions. For the case of log-normally distributed fracture lengths, also a user defined standard deviation σ_{fr} has to be specified. In the case of gamma distributed fracture lengths, the only needed parameter to characterize the distribution are the user defined shape k and scale θ. Three examples of originated fields are reported in this section in Fig 5.2, 5.3, 5.4; for better visibility in the plots the log-hydraulic conductivity of the fields was set to zero (homogeneous fields). The framework utilize half fracture length as inputs, therefore for better readability this notation is used in the examples.

The new definition of effective segment which also considers the high conductivity regions in the porous matrix has been implemented. The originated pathlines and effective data points can be seen in Fig. 5.1. This addition to the framework should enable the data driven SDM to be applied to fields with the following characteristics: very low fracture density $f < 3\%$ especially with fractures aligned with mean pressure gradient, and/or poorly conductive fractures (obstacles). In Fig. 5.1 two fields realizations with a particle pathline are shown. To improve the visibility of the high conductivity regions a field variance value of $\sigma_Y^2 = 4$ was chosen and the colorbar maximum value was set equal to the threshold used to define the high conductivity regions; therefore all the regions with conductivity equal or higher than 2 are shown as black. Fractures are represented as black horizontal thick lines with $ln(K) = 4$. In the upper field with the classical approach, the tracked particle does not enter any fracture and therefore would lead to the definition of zero effective segments. With the new definition it is possible to define some effective segments. In the lower plot it is shown that effective data points are defined at both fractures and porous matrix high conductivity region exit points. Due to the time and resources

limitations of this project, I was not able to check the effect that the new definition of effective segments has on different medium characterization. The only drawback of this system is the possible increase in computational time for bigger fields. Extensive testing on different medium characterizations should be carried out in order to find when this "extended definition" is useful and when it does not.

To extend the definition from cell size region to multiple cell sized regions and therefore set the conductivity threshold, the correlation length l_Y should be considered. This would allow to obtain effective segments in the matrix where the Lagrangian pathline parameters are correlated along the same segment, similar to what is currently happening in fractures. This would allow each effective segment to define a proper pathline state.

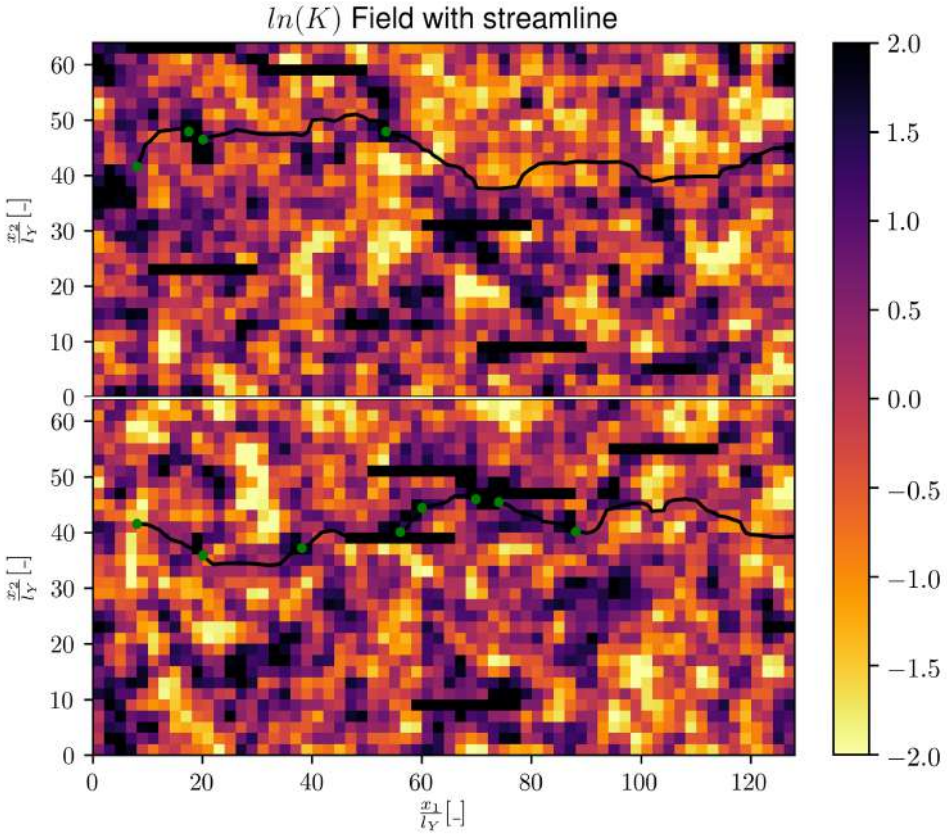


Figure 5.1.: Two log-hydraulic conductivity fractured porous field realizations generated with the IFD framework, 1 pixel line fracture family: $\theta_{fr} = 0°$, $l_{fr} = 20$, $a_{fr} = 2$, the domain resolution is 64×32 cells with size 128×64. The correlation length is 1 cell in x,y for both directions and $\sigma_Y^2 = 4$. The green dots along the black particle pathline represent the effective data points and are generated directly from the code. Logarithmic colorbar scaling. In the plot above the main advantage of the new extended definition is displayed; the particle pathline does not enters any fracture but still produce effective data points in the porous matrix.

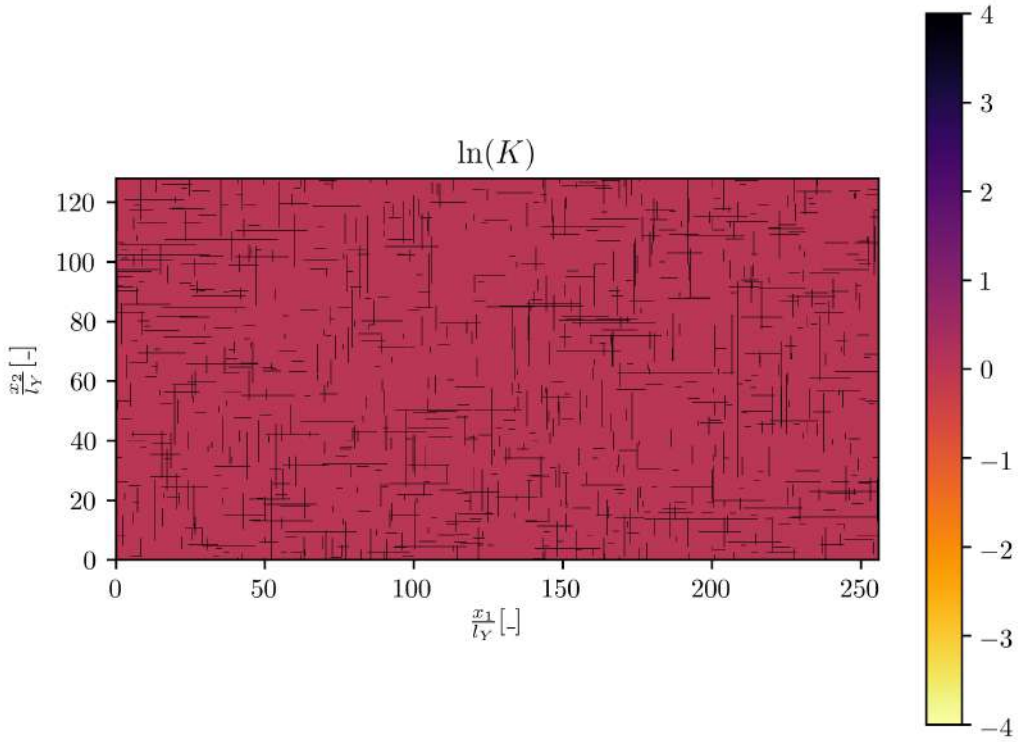


Figure 5.2.: Log-hydraulic conductivity fractured porous field map generated with the IFD framework, 2 pixel lines fracture families: $\theta_{fr} = 90°, 0°$, exponential distributions mean $l_{fr} = 3$, fractures half length $x \sim \text{Exp}(1/l_{fr})$, fractures aperture $a_{fr} = 0.25$, the domain resolution is 1024×512 cells. The correlation length is 4 cells in x,y for both directions and $\sigma_Y^2 = 0$, fracture density $f = 5\%$.
The PDF reads as follows: $p(x) = \frac{1}{l_{fr}} \cdot \exp\left[-\frac{x}{l_{fr}}\right] \quad \forall x \geq 0$

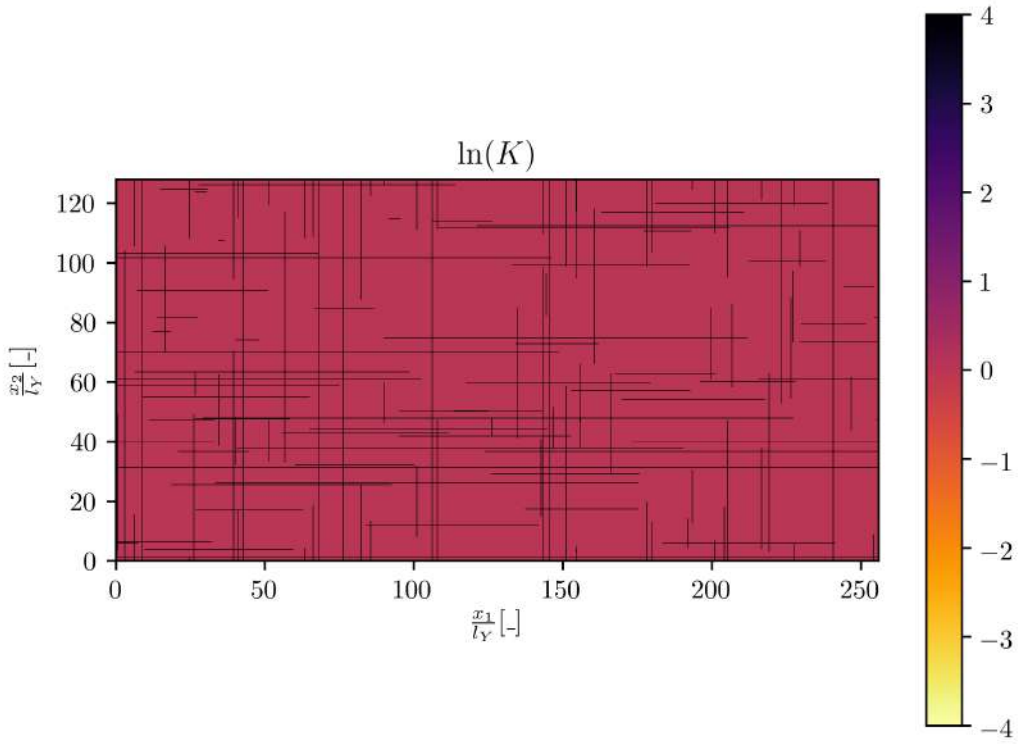


Figure 5.3.: Log-hydraulic conductivity fractured porous field map generated with the IFD framework, 2 pixel lines fracture families: $\theta_{fr} = 90°, 0°$, log-normal distributions mean $l_{fr} = 3$, distributions standard deviation $\sigma_{fr} = 1$, fractures half length $\ln(x) \sim \mathcal{N}(l_{fr}, \sigma^2_{fr})$, fractures aperture $a_{fr} = 0.25$, the domain resolution is 1024×512 cells. The correlation length is 4 cells in x,y for both directions and $\sigma^2_Y = 0$, fracture density $f = 5\%$. The PDF reads as follows: $p(x) = \frac{1}{x\sigma_{fr}\sqrt{2\pi}} \cdot \exp\left[-\frac{(lnx - l_{fr})^2}{2\sigma^2_{fr}}\right]$

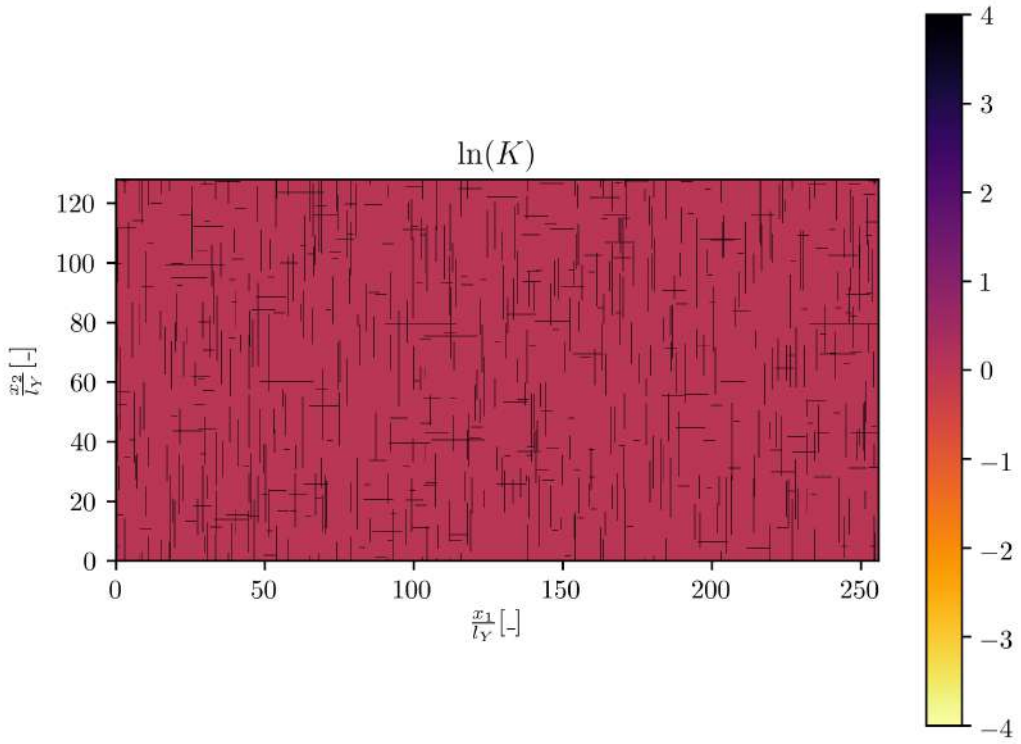


Figure 5.4.: Log-hydraulic conductivity fractured porous field map generated with the IFD framework, 2 pixel lines fracture families: $\theta_{fr} = 90°, 0°$, gamma distribution shape $k = 7.5, 1$, scale $\theta = 1.0, 2.0$, fractures half length $x \sim \Gamma(k, \theta)$, fractures aperture $a_{fr} = 0.25$, the domain resolution is 1024×256 cells. The correlation length is 4 cells in x,y for both directions and $\sigma_Y^2 = 0$, fracture density $f = 5\%$.

The PDF reads as follows: $p(x) = \frac{x^{k-1}}{\Gamma(k)\theta^k} \cdot exp\left[-\frac{x}{\theta}\right] \quad \forall x, k, \theta > 0 \quad ; \quad \Gamma(k) = (k-1)!$

6. Conclusions

In this work, the generalized correlated random walk model for 2-D macrodispersion in fractured porous media, developed at IFD, was tested for different random fractured medium characterizations. Specifically, fields with varied fracture density and log-conductivity variance were considered in combination with different fractures orientations. These features were chosen because of the significant influence they have on dispersion and on the model performance.

It has been observed that the model is able to predict the asymptotic longitudinal and transverse spreading behaviour of a solute plume in fields having conductivity variances ranging in the interval $\sigma_Y^2 \in [1/16, 4]$. The model is also applicable for fields with multiple fracture families and fracture densities in the range $f \in [3\%, 20\%]$. Increasing the fracture density and log-conductivity variance has been shown to lead to a slight increase in inaccuracy of the model. The reason behind this behaviour is still unknown, but it is thought to be related to the simplified structure of the model which does not sufficiently capture the prolonged correlation in the effective particle motion. The fracture density bounds can be extended further by using an alternative definition of effective segments which is proposed in this thesis. In this definition, the effective pathline segment is defined between transition points from a high conductivity region to a low conductivity region along the particle trajectory. It uses all high conductivity regions, including those in the matrix, as effective data points. The possibility to generate fields with exponentially or log-normally distributed fracture lengths has been implemented.

Further extensions to the IFD framework should include finding new field variables upon which the dispersion process is dependent and, thus, reformulating the transition probabilities of the correlated random walk model. This could be achieved by using a neural network to find new correlations between the particle trajectories and field parameters. The creation of fractures with fractal geometry and variable aperture.

Acknowledgements: I would like to thank Ranit Monga, the supervisor of this thesis, for his patience in introducing me to this vast field of stochastic computational methods and for his constant fast help, support and care. I'm also grateful to be able to have worked in the Prof. Patrick Jenny's group at the Institute of Fluid Dynamics. All the computations were performed with the Euler HPC cluster of ETH Zürich.

Bibliography

[1] Bakr Adel A. *et al.*1978.STOCHASTIC ANALYSIS OF SPATIAL VARIABILITY IN SUBSURFACE FLOWS 1. COMPARISON OF ONE- AND THREE- DIMENSIONAL FLOWS.Water Resources Research 14(2).

[2] Berkowitz Brian, Levy Melissa.2002.MEASUREMENT AND ANALYSIS OF NON-FICKIAN DISPERSION IN HETEROGENEOUS POROUS MEDIA.Journal of Contaminant Hydrology 64(2003).

[3] Bijeljic Branko, Blunt Martin J..2006.PORE-SCALE MODELING AND CONTINUOUS TIME RANDOM WALK ANALYSIS OF DISPERSION IN POROUS MEDIA.Water Resources Research 42(W01202).

[4] Bonnet E. *et al.*2001.SCALING OF FRACTURE SYSTEMS IN GEOLOGICAL MEDIA.Reviews of Geophysics 39(3).

[5] Bogdanov I. I. *et al.*2007.EFFECTIVE PERMEABILITY OF FRACTURED POROUS MEDIA IN STEADY STATE FLOW.Water Resources Research 39(1).

[6] Brenner Oliver.2016.GENERALIZED CTRW APPROACH FOR FRACTURED HETEROGENEOUS MEDIA. Master Thesis ETH Zurich.

[7] Dagan Gedeon.1987.THEORY OF SOLUTE TRANSPORT BY GROUNDWATER.Annual Review of Fluid Mechanics 19(1987).

[8] Duguid James O., Lee P. C. Y..1977.FLOW IN FRACTURED POROUS MEDIA.Water Resources Research 13(3).

[9] Hakami Eva.1995.APERTURE DISTRIBUTION OF ROCK FRACTURES. Doctoral Thesis Royal Institute of Technology Stockholm.

[10] Jenny Patrick *et al.*2007.COMPUTING LIGHT STATISTICS IN HETEROGENEOUS MEDIA BASED ON A MASS WEIGHTED PROBABILITY DENSITY FUNCTION METHOD.Journal of the Optical Society of America 2206-2219(24).

[11] Kang Peter K. *et al.*2015.ANOMALOUS TRANSPORT ON REGULAR FRACTURE NETWORKS: IMPACT OF CONDUCTIVITY HETEROGENEITY AND MIXING AT FRACTURE INTERESCETIONS.Physical Review E 92(022148).

[12] Kubeyev Amanzhol.2013.IMPACT OF MULTI-SET FRACTURE PATTERN ON THE EFFECTIVE PERMEABILITY OF FRACTURED POROUS MEDIA.Journal of Petroleum Science and Engineering 161-169 (112).

[13] Le Borgne T. *et al.*2008.LAGRANGIAN STATISTICAL MODEL FOR TRANSPORT IN HIGHLY HETEROGENEOUS VELOCITY FIELDS.Physical Review Letters 101(090601).

[14] Le Borgne T. *et al.*2008.SPATIAL MARKOV PROCESSES FOR MODELING LAGRANGIAN PARTICLE DYNAMICS IN HETEROGENEOUS POROUS MEDIA.Physical Review E 78(026308).

[15] Le Borgne T. *et al.*2011.EFFECTIVE PORE-SCALE DISPERSION UPSCALING WITH A CORRELATED CONTINUOUS TIME RANDOM WALK APPROACH.Water Resources Research 47(W12538).

[16] Martin Vincent *et al.*2005.MODELING FRACTURES AND BARRIERS AS INTERFACES FOR FLOW IN POROUS MEDIA.SIAM Journal on Scientific Computing 26(5).

[17] Meyer Daniel W. *et al.*2010.A JOINT VELOCITY-CONCENTRATION PDF METHOD FOR TRACER FLOW IN HETEROGENEOUS POROUS MEDIA . Water Resources Research 46(W12522).

[18] Meyer Daniel W., Tchelepi Hamdi A..2010.PARTICLE-BASED TRANSPORT MODEL WITH MARKOVIAN VELOCITY PROCESSES FOR TRACER DISPERSION IN HIGHLY HETEROGENEOUS POROUS MEDIA.Water Resources Research 46(W11552).

[19] Meyer Daniel W.*et al.*2013.A FAST SIMULATION METHOD FOR UNCERTAINTY QUANTIFICATION OF SUBSURFACE FLOW AND TRANSPORT.Water Resources Research 49(5).

[20] Monga Ranit.2018.THEORETICAL INVESTIGATION AND MODELING OF DISPERSION IN RANDOM POROUS MEDIA. Master Thesis ETH Zurich.

[21] Monga Ranit *et al.*2019.A CORRELATED RANDOM WALK MODEL FOR MACRODISPERSION IN FRACTURED POROUS MEDIA HAVING BI-MODAL CONDUCTIVITY DISTRIBUTION.Submitted to Water Resources Research.

[22] [a].Naff R. L. *et al.*1998.HIGH-RESOLUTION MONTE CARLO SIMULATION OF FLOW AND CONSERVATIVE TRANSPORT IN HETEROGENEOUS POROUS MEDIA 1.METHODOLOGY AND FLOW RESULTS.Water Resources Research 34(4).
[b].Naff R. L. *et al.*1998.HIGH-RESOLUTION MONTE CARLO SIMULATION OF FLOW AND CONSERVATIVE TRANSPORT IN HETEROGENEOUS POROUS MEDIA 2.TRANSPORT RESULTS.Water Resources Research 34(4)

[23] Saladin Paolo, Fiorotto Virgilio.1998.SOLUTE TRANSPORT IN HIGHLY HETEROGENEOUS AQUIFERS.Water Resources Research 34(5).

[24] Simmons Craig T. *et al.*2001.VARIABLE-DENSITY GROUNDWATER FLOW AND SOLUTE TRANSPORT IN HETEROGENEOUS POROUS MEDIA:APPROACHES,RESOLUTION AND FUTURE CHALLENGES.Journal of Contaminant Hydrology 51(2001).

[25] Suidicky E. A., Frind E. O..1982.CONTAMINANT TRANSPORT IN FRACTURED POROUS MEDIA:ANALYTICAL SOLUTIONS FOR A SYSTEM OF PARALLEL FRACTURES.Water Resources Research 18(6).

[26] The Free Encyclopedia Wikipedia.2020.MULTIGRID METHOD.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Multigrid_method (Accessed 06.04.2020)

[27] The Free Encyclopedia Wikipedia.2020.MONTE CARLO METHOD.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Monte_Carlo_method (Accessed 16.03.2020)

[28] The Free Encyclopedia Wikipedia.2020.MARKOV CHAIN.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Markov_chain (Accessed 20.03.2020)

[29] The Free Encyclopedia Wikipedia.2020.STOCHASTIC MATRIX.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Stochastic_matrix (Accessed 20.03.2020)

[30] The Free Encyclopedia Wikipedia.2020.EMPIRICAL DISTRIBUTION FUNCTION.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Empirical_distribution_function (Accessed 09.04.2020)

[31] The Free Encyclopedia Wikipedia.2020.DARCY LAW.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Darcy%27s_law (Accessed 12.03.2020)

[32] The Free Encyclopedia Wikipedia.2020.INVERSE TRANSFORM SAMPLING.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Inverse_transform_sampling (Accessed 09.04.2020)

[33] The Free Encyclopedia Wikipedia.2020.POROUS MEDIUM.Wikimedia foundation. URL https://en.wikipedia.org/wiki/Porous_medium (Accessed 28.02.2020)

[34] Tibbit Mark.2020.MACROMOLECULAR ENGINEERING:NETWORKS AND GELS.Lecture Notes.

[35] Zhang Dongxiao, Kang Quinjun.2004.PORE SCALE SIMULATION OF SOLUTE TRANSPORT IN FRACTURED POROUS MEDIA .Geophysical Research Letter, 31(L12504).

[36] Bijeljic Branko *et al* .2013.PREDICTIONS OF NON-FICKIAN SOLUTE TRANSPORT IN DIFFERENT CLASSES OF POROUS MEDIA USING DIRECT SIMULATION ON PORE-SCALE IMAGES.Physical Review E 87(013011. DOI: https://doi.org/10.1103/PhysRevE.87.013011).

A. Fractured porous fields

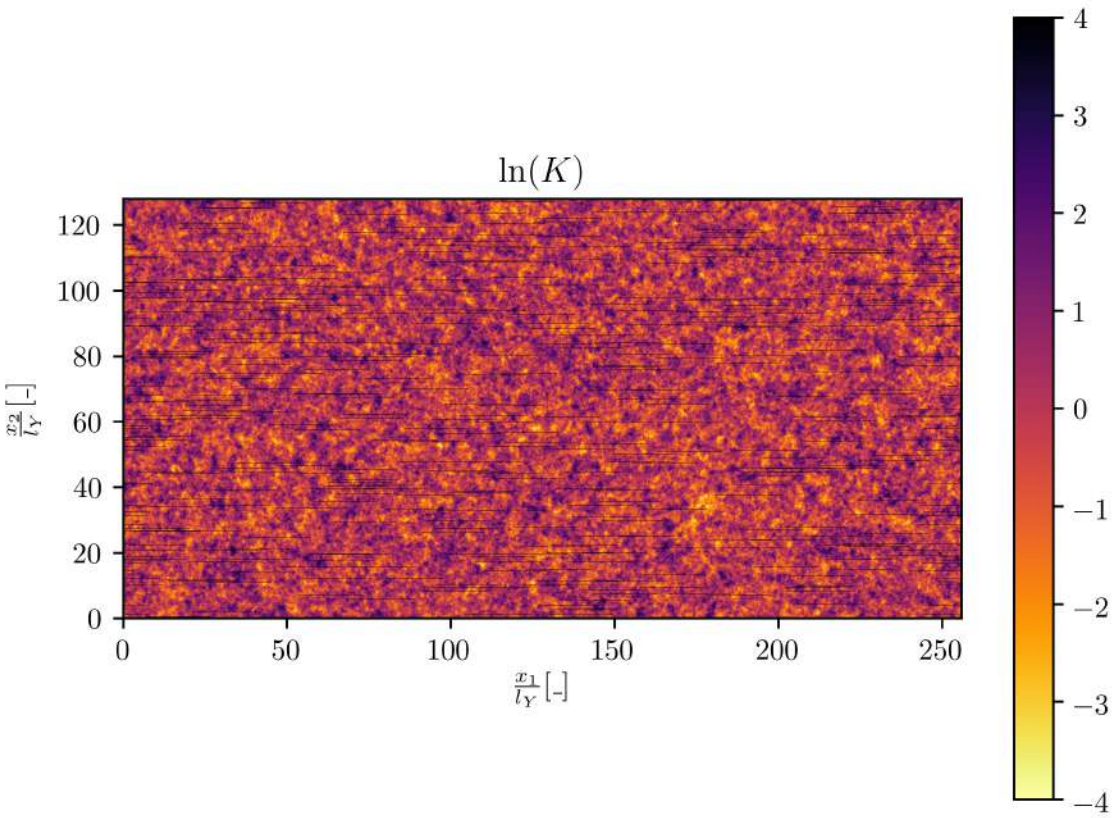


Figure A.1.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 1$, $f = 3\%$.

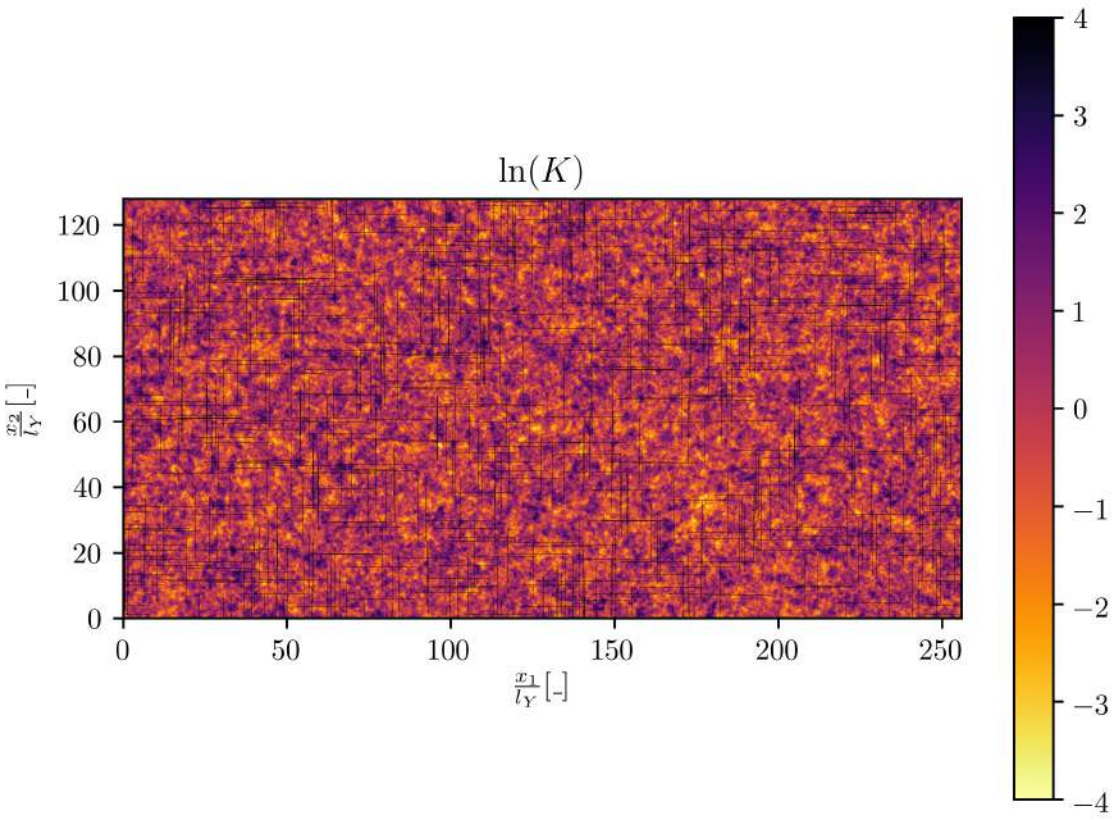


Figure A.2.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 1, f = 5\%$.

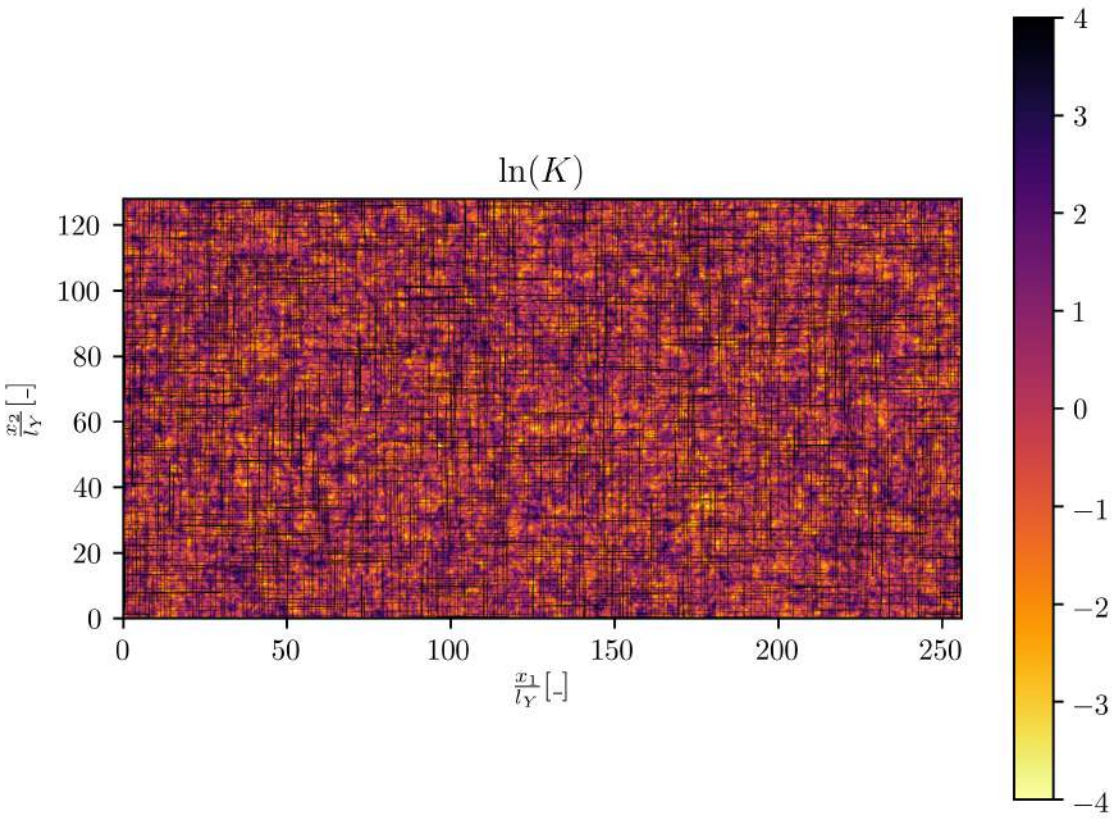


Figure A.3.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 1$, $f = 20\%$.

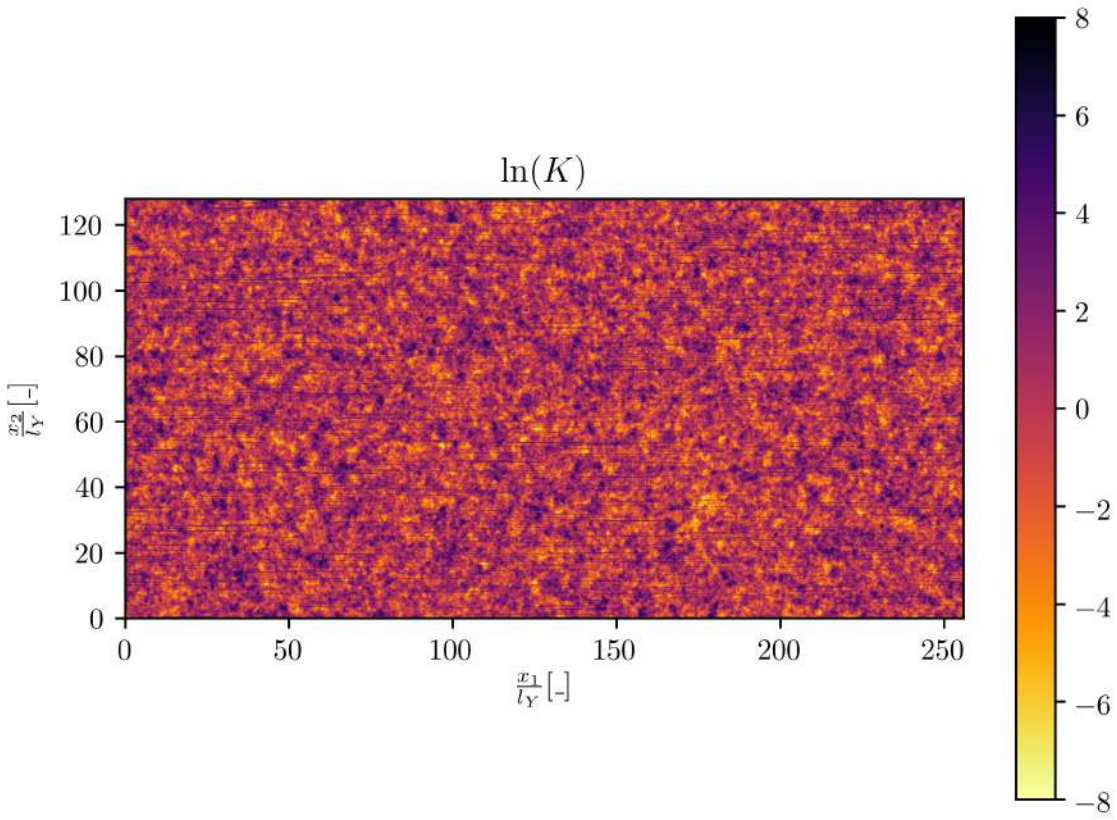


Figure A.4.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 4$, $f = 5\%$, $\theta_{fr} = 0°$.

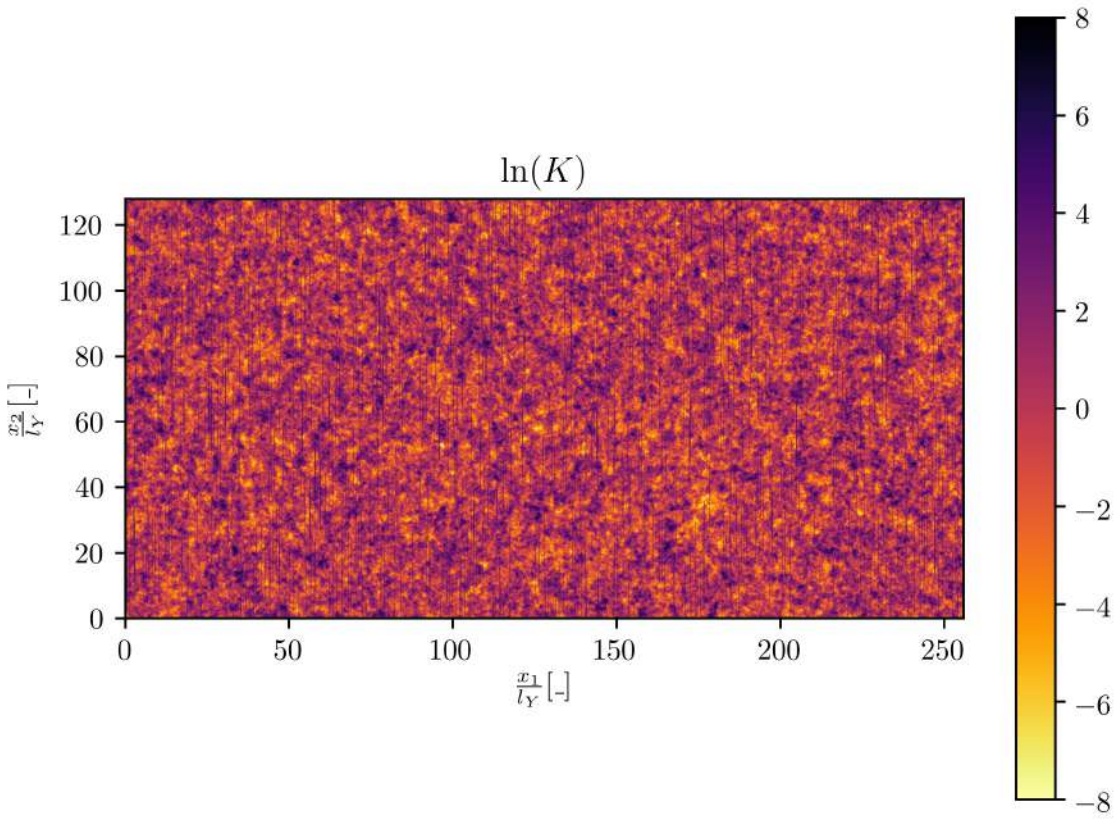


Figure A.5.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 4$, $f = 5\%$, $\theta_{fr} = 90°$.

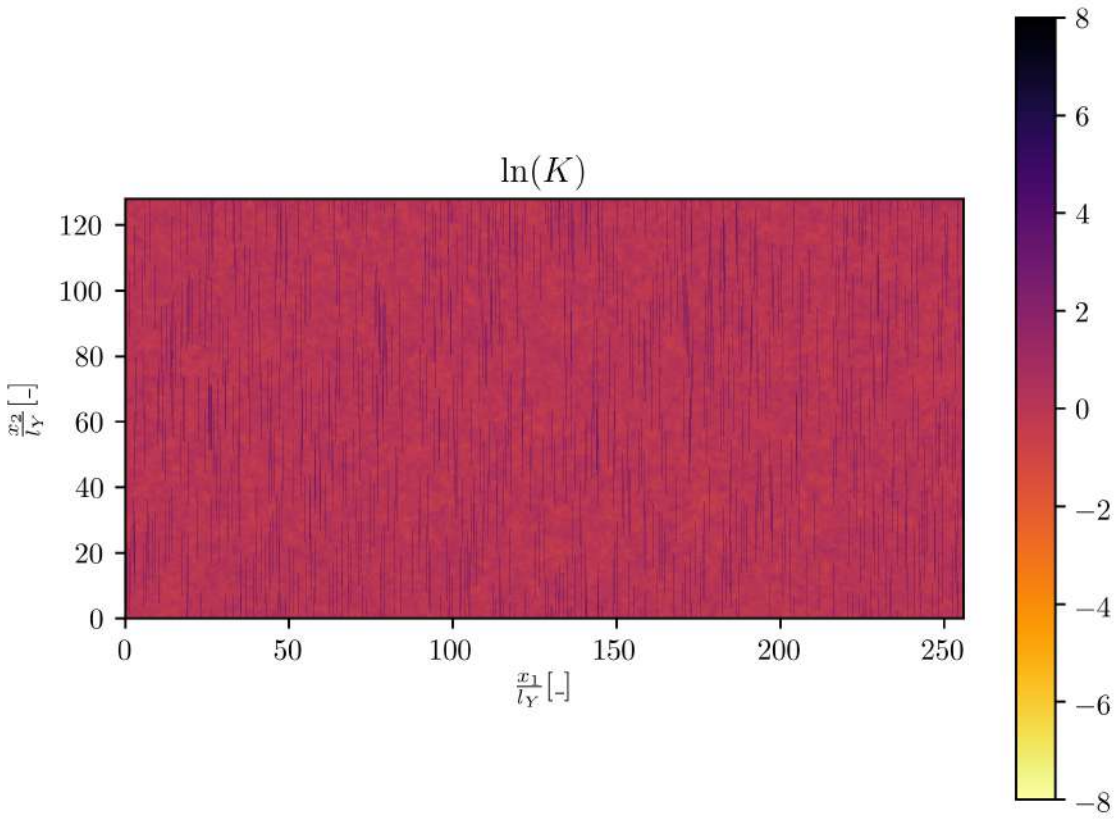


Figure A.6.: A fractured porous field realization from MCS of new tested medium characterization: fracture density case, $\sigma_Y^2 = 1/16$, $f = 5\%$.

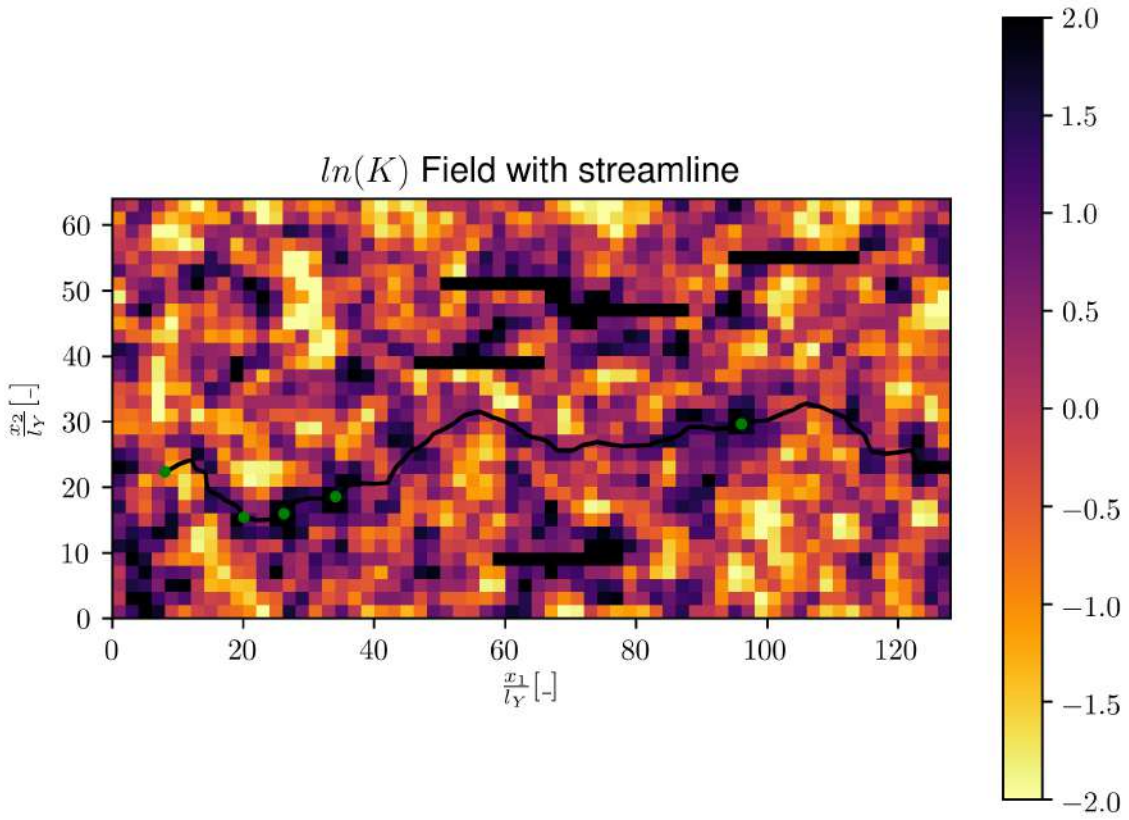


Figure A.7.: A fractured porous field realization from MCS with the new extended definition of effective segment, same settings as the one showed in Fig. 5.1.

B. IFD code framework improvements

The code is subdivided in several different files containing specific code functionalities (classes definitions, functions,...). Everything is called from the main file `example_parallel.py`. This file is responsible for taking the simulation parameters from `test.par` and passing them to the other functions. The bi-modal fractured porous field generation and pathline creation and chopping is carried out in `streamline_data_generator.py`, most of the modification were applied to this file, while in the other file the new parameterss were defined. The following code are extracts of the whole code and contains the modifications I applied; the notation "..." indicates that the code continues but the content is not written. For the fracture functions, the base code was taken from the already present function `pixel_lines()`, therefore the missing lines are intended to be the same as the one present there. To enable random sampling from a PDF the Numpy packed random has been added to `streamline_data_generator.py`:

```
import numpy as np
import math

from numpy import random
from scipy.special import erf, erfinv
...
```

The following lines are used to enable the calling of the new defined functions:

```
...
class FieldGenerator:
    """

    A field generator for 2D multi-Gaussian fields, and transformations,
    with constant properties.

    """

    def __init__(self, nx, ny, lx, ly, field_type, m, cnx, cny, lx_e,
                 ly_e, la_e, sigma_length, MAX_COV, weight_Gauss,
                 weight_Fractures, seed=None, dtype='float64', order='F'):
    """
    ...
        :param field_type: Field type:
        (0): MultiGaussian field with pixel line fractures;
        (1): MultiGaussian with elliptical fractures;
        (2): Zinn Disconnected field with line fractures;
        (3): Zinn Disconnected field with elliptical fractures;
        (4): Zinn Connected field with line fractures;
        (5): Zinn Connected field with elliptical fractures
        (6): MultiGaussian field with exponentially distributed
        pixel line fractures lengths
        (7) MultiGaussian field with pixel line fracture with lognormally
        distributed lengths
        :type field_type: int
    ...
    """
        self.lx_e = lx_e
        self.ly_e = ly_e
        self.la_e = la_e
```

```
        self.sigma_length = sigma_length
        self.MAX_COV = MAX_COV
    ...

    def run(self):
        """


        Copy of the Matlab field generator fieldGeneratorMultiGaussian.m by
        Daniel W. Meyer.

        """
        self.x_i, self.y_i = np.meshgrid(np.arange(self.nx), np.arange(self.ny))

        self.xc = 0.5 * (self.nx - 1.0)
        self.yc = 0.5 * (self.ny - 1.0)

        if (self.field_type == 0) or
        (self.field_type==1) or (self.field_type == 6) or (self.field_type == 7):

            self.field_values = self.MultiGaussian()

        if (self.field_type == 2) or (self.field_type==3):

            self.field_values = self.ScoreTransformDisconnected()

        if (self.field_type == 4) or (self.field_type==5):

            self.field_values = self.ScoreTransformConnected()

        self.field_values = self.weight_Gauss*self.field_values

        if (self.field_type == 0) or (self.field_type==2) or
        (self.field_type==4):
            self.frac_locations, self.frac_index_dict = self.pixel_lines()
        if (self.field_type == 6):
            self.frac_locations, self.frac_index_dict = self.pixel_lines_exp()
        if (self.field_type == 7):
            self.frac_locations, self.frac_index_dict = self.pixel_lines_lognorm()
        if (self.field_type == 1) or (self.field_type==3) or (self.field_type==5):
            self.frac_locations, self.frac_index_dict = self.Ellipses()
    ...
```

The following lines are used to define exponentially distributed fracture lengths:

```
...
    def pixel_lines_exp(self):


        while coverage < self.MAX_COV:

        ...
                for ii in np.arange(-n_p, n_p + 1.0):
                    for jj in np.arange(-n_p, n_p + 1.0):
                        X_curr = (X + ii * self.lx - xe)
                        Y_curr = (Y + jj * self.ly - ye)
                        frac = np.multiply((np.absolute(np.multiply(X_curr,
                                np.cos(self.la_e[n])) + np.multiply(Y_curr,
                                np.sin(self.la_e[n]))) <=
                                int(np.random.exponential(scale = self.lx_e[n] ,
                                size= None))),(np.absolute(-1.0*np.multiply(X_curr,
                                np.sin(self.la_e[n])) + np.multiply(Y_curr,
```

```
                                np.cos(self.la_e[n]))) <=self.ly_e[n]))
                                ...
        ...
...
```

The following lines are used to define log-normally distributed fracture lengths:

```
    def pixel_lines_lognorm(self):


        while coverage < self.MAX_COV:

        ...
                for ii in np.arange(-n_p, n_p + 1.0):
                    for jj in np.arange(-n_p, n_p + 1.0):
                        X_curr = (X + ii * self.lx - xe)
                        Y_curr = (Y + jj * self.ly - ye)
                        frac = np.multiply((np.absolute(np.multiply(X_curr,
                                np.cos(self.la_e[n])) + np.multiply(Y_curr,
                                np.sin(self.la_e[n]))) <=
                                int(np.random.lognormal(mean = self.lx_e[n] ,
                                sigma = int(self.sigma_length[n]) ,
                                size= None))), (np.absolute(-1.0*np.multiply(X_curr,
                                np.sin(self.la_e[n])) + np.multiply(Y_curr,
                                np.cos(self.la_e[n]))) <= self.ly_e[n]))
                                ...

...
```

The following lines are used to define gamma distributed fracture lengths:

```
    def pixel_lines_gamma(self):


        while coverage < self.MAX_COV:
         ...
                for ii in np.arange(-n_p, n_p + 1.0):
                    for jj in np.arange(-n_p, n_p + 1.0):
                        X_curr = (X + ii * self.lx - xe)
                        Y_curr = (Y + jj * self.ly - ye)
                        frac = np.multiply((np.absolute(np.multiply(X_curr,
                                np.cos(self.la_e[n])) + np.multiply(Y_curr,
                                np.sin(self.la_e[n]))) <=
                                int(np.random.gamma(self.shape_k[n] ,
                                self.scale_theta[n], size= None))),
                                (np.absolute(-1.0*np.multiply(X_curr,
                                np.sin(self.la_e[n])) + np.multiply(Y_curr,
                                np.cos(self.la_e[n]))) <= self.ly_e[n]))

...
```

The following lines are used to implement the extended definition of effective segments, where all the high conductivity regions exit points (fractures and porous matrix) are considered as possible effective data points; the explanation is contained in the code comments:

```
...
def split_streamline_spatial_matrix_fracture(streamline,
                                             hydraulic_conductivity_field, x_term):
    ...
    # number of effective segments
    d_cross = None
```

```
# whether the first effective segment starts in a fracture or in matrix
first_seg = None

#the variables that define the cell hydraulic conductivity,
#TODO:import them from test.par file or in test.par insert new parameter:
#"ln(K)_treshold"

#these parameters can be used to define the conductivity threshold
#if used, uncomment the line of the threshold
sigma_y          = 2
weight_Fractures = 2

#fracture generation produced a dictionary containing all cells linear index as keys
#and as value the fracture number, if any, present in that cell.

#The overall goal is to isolate the matrix high conductivity regions and add them to
#the dictionary at the end.

#To do that it is useful to clean the fracture index dictionary and remove the cells
#where no fracture is present.
#frac_index_dict has the form {cell number i:[fracture k, fracture m,...],...}

clean_frac_index_dict = hydraulic_conductivity_field.frac_index_dict.copy()
for k, v  in clean_frac_index_dict.items():
    if len(v) == 0:
        del clean_frac_index_dict[k]

#Im interested in the index of the cells contaning fractures,
#therefore the keys are excracted

keys_list = list(clean_frac_index_dict.keys())

#each new found conducitivty matrix region will have its own number, as it
#were a fracture (cell size fracture).
#To know the last fracture number used (placed) i need the values of the
#fracture indeces dictionary

values_list = list(hydraulic_conductivity_field.frac_index_dict.values())

#now i create an array which contains the x,y indices of all the cells
#that satisfy the high conducitivty region condition
#domain_field_indeces has the form [[x_1,x_38;....] , [y_1,y_38,...]]
#following this definition also the cells with fractures are considered,
#this is not a problem since we know which one they are from the keys list

domain_field_indeces = np.nonzero(hydraulic_conductivity_field.values >= 2)
#the treshold here is put to two, otherwise parameters can be used, but
#they are not imported from test.par
#domain_field_indeces = np.nonzero(hydraulic_conductivity_field.values >=
#                          sigma_y * weight_Fractures)
#now a linear index is assigned to each high conductivity regions and the
#numbers goes until the max number of cells

domain_field_indeces_linear = np.ravel_multi_index(domain_field_indeces,
                              (hydraulic_conductivity_field.values.shape[0]+1,
                              hydraulic_conductivity_field.values.shape[1]+1),
                              order='C')


#now the cells indeces present in domain_field_indeces_linear that are not
# present in keys_list are extracd, i.e the matrix high cond regions indeces
```

```
    matrix_high_cond_regions_indeces_linear=list(np.setdiff1d(domain_field_indeces_linear
                                            , keys_list))


    #the last fracture number sued by the fracture generator is extracted

    last_frac_number = np.amax(np.amax(values_list))+1

    #the counter m is created to iterate over the new found high conductivity
    #regions cells linear indices, which are keys  of the fracture dictionary
    #but with empty value. Now a value is added and correspond do a cell size
    #fracture number starting from last_frac_number
    #this procedure can be done also with a for loop.
    #Also update should work but append ended up working better for me

    m = 0
    while(m<len(matrix_high_cond_regions_indeces_linear)):
        h = matrix_high_cond_regions_indeces_linear[m]
        hydraulic_conductivity_field.frac_index_dict[h].append(last_frac_number)
        #now the counters are updated/iterated before nexxt cycle start
        m += 1
        last_frac_number += 1


    #after this while loop the frac_index_dict has been updated with the new high
    #conductivity regions indices at the right cell linear index
    #and therefore the high conductivity regions will be interpreted as
    #fractures




    # linear index of the first point, used as the key to dictionary.
    # To identify the if the cell and streamline point
    # belong to a fracture


    ...

...
```

In `example_parallel.py` the following lines were inserted (or modified) along the existing ones:

```
...
    :param args: A list containing all required arguments in the prescribed \
    order.
    :type args: list
    """

    data_path, case_id, nx, ny, lx, ly, usl_x, usl_x_term,usl_y1, usl_y2,\
        qh_0_i2, h_n1p1_i2, h2_over_h1, qhp, sigma_y, tol, field_type, m, cnx,\
        cny, lx_e, ly_e, la_e, sigma_length, shape_k, scale_theta, MAX_COV, weightGauss,
        weightFractures, step_size, passes, max_number_vertices, splitter_length,
         splitter_time, n_sl,write_xyt, generate_plots,
         write_fields, n_fields, child_seeds, i_field = args
```

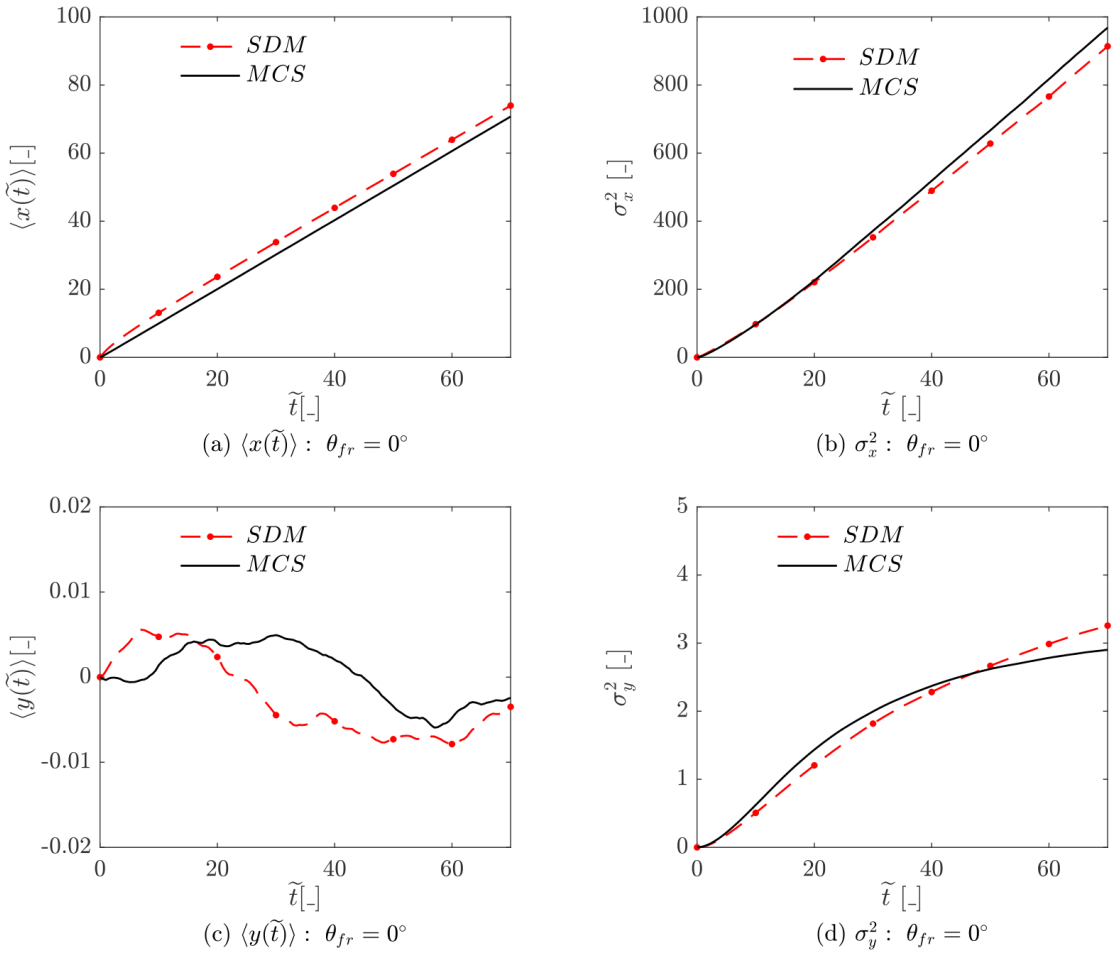
```
...
    # field generator
    field_generator = FieldGenerator(nx=nx, ny=ny, lx=lx, ly=ly, field_type=field_type,
                                     m=m, cnx=cnx, cny=cny, lx_e=lx_e, ly_e=ly_e,
                                     la_e=la_e, sigma_length=sigma_length,
                                     shape_k = shape_k, scale_theta = scale_theta,
                                     MAX_COV=MAX_COV, weight_Gauss=weightGauss,
                                     weight_Fractures=weightFractures,
                                      seed=child_seeds[2*i_field])
...

    # Exponential (True) or Gaussian covariance (False)
    m = parameter_data.get_bool(par='m')

    # geometrical parameters for the elliptical fractures, if any
    nValues = parameter_data.get_n_values('field_lx_e')
    assert (nValues == parameter_data.get_n_values('field_ly_e'))
    assert (nValues == parameter_data.get_n_values('field_la_e'))
    assert (nValues == parameter_data.get_n_values('sigma_length'))
    assert (nValues == parameter_data.get_n_values('shape_k'))
    assert (nValues == parameter_data.get_n_values('scale_theta'))

    lx_e = np.zeros(nValues)
    ly_e = np.zeros(nValues)
    la_e = np.zeros(nValues)
    sigma_length = np.zeros(nValues)
    shape_k = np.zeros(nValues)
    scale_theta = np.zeros(nValues)

    # accessing the geometrical parameters of each fracture family
    for i in xrange(nValues):
        lx_e[i] = parameter_data.get_num('field_lx_e', i)
        ly_e[i] = parameter_data.get_num('field_ly_e', i)
        la_e[i] = parameter_data.get_num('field_la_e', i)
        sigma_length[i] = parameter_data.get_num('sigma_length', i)
        shape_k[i] = parameter_data.get_num('shape_k', i)
        scale_theta[i] = parameter_data.get_num('scale_theta', i)

...

    # create parameter list
    args = [data_path, case_id, nx, ny, lx, ly, usl_x, usl_x_term, usl_y1, usl_y2,
            qh_0_i2, h_n1p1_i2, h2_over_h1, qhp,
            sigma_y, tol, field_type, m, cnx, cny, lx_e, ly_e, la_e, sigma_length,
            shape_k, scale_theta, MAX_COV,
            weightGauss, weightFractures, step_size, passes,
             max_number_vertices, splitter_length, splitter_time,
            n_sl, write_xyt, generate_plots, write_fields,
            n_fields, child_seeds]
```

C. Field variance case: $\sigma_Y^2 = 1/16$, $f = 5\%$, $\theta_{fr} = 0°$

The Stochastic Simulations (SS) of this case (multiple were ran) produced "corrupted" files or could not proceed (crashed) after few time steps. The SS were at first ran on ETH Euler HPC cluster as all the other simulations. On the cluster the first run kept producing clean data for some dozen of time steps, after which the data started being duplicated (always the same as previous step) for thousand of time steps of time steps forward. After a certain time the produced data would go back to the last non corrupted time step and restart normally from there, and after other dozen time steps start to loop again. Successive runs were made, which were stable, and each one produced data until a different time step (not enough to reach $\widetilde{t} = 200$). Apart from the duplicate data, all the other data in all the four runs was the same. Due to security reasons in may 2020, the Euler cluster was not accessible for further computations. The Transition Matrices produced by the past simulation were stored elsewhere and thus accessible. I moved the Stochastic Simulation on IFD computers. Several more runs were attempted but each time they would crash after time step 17.

My hypothesis is that the simulations are giving errors due to the quite homogeneous nature of the tested case. The tracer particles are travelling mostly into the longitudinal fractures and the outside not very heterogeneous matrix does not impact their trajectory much; therefore the produced states used in the TMs are very similar between each others and end up in very few bins. When the program try to extract a new state, it encounters many empty bins and thus crashes. The difference between the reached time step between the simulations and different machines could be caused by how random numbers (for state selection) are generated. This hypothesis could be tested by checking the data containers (they contains the data used for the TM construction), but unfortunately they are stored on the Euler cluster and the time for this project has finished.

The plots are generated using the reference MCS and the non duplicated data extracted from the first SS run. In case in future this problem is solved and the obtained data will be the same as mine, then these plots are correct and usable; otherwise I cannot guarantee the correctness of the data reported below and thus should not be used. Other plots could not be generated due to the needed files being corrupted and not usable.



(a) $\langle x(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (b) σ_x^2 : $\theta_{fr} = 0°$

(c) $\langle y(\widetilde{t})\rangle$: $\theta_{fr} = 0°$ (d) σ_y^2 : $\theta_{fr} = 0°$

Figure C.1.: Mean and variance in longitudinal (above) and transverse (below) direction. Field fracture density 5% with log-hydrualic conductivity variance $\sigma_Y^2 = 1/16$. Plotted are the data produced by the Stochastic Dispersion Model (SDM) and the reference Monte Carlo Simulation (MCS). The data is plotted until the first particle exit time in MCS: $\widetilde{t} = 70$

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

STOCHASTIC MODELING OF DISPERSION IN FRACTURED POROUS MEDIA

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
FABIANO	SASSELLI

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zurich, 4.20.2020	Fabiano Sasselli

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.