Three-dimensional simulation of tracer transport dynamics in formations with high-permeability channels or fractures: Estimation of oil saturation

Siarhei Khirevich[1, a] and Tadeusz W. Patzek[1]
Ali I. Al-Naimi Petroleum Engineering Research Center, King Abdullah University of Science and Technology, 23955-6900, Thuwal, Saudi Arabia

(Dated: 15 November 2019)

We simulate flow and dispersion of tracers in three-dimensional fractured geometries obtained with Voronoi tessellations. "Fractures" are generated and discretized using a parallel in-house code. These "fractures" can also be regarded as the high-permeability flow paths through the rock, or a network of the "super-k" channels. Generated geometry contains multiply-connected matrix and fracture regions. The matrix region represents a porous rock filled with solid, water, and oil. Tracers diffuse in both regions, but advection is limited only to the fractures. The lattice-Boltzmann and random-walk particle-tracking methods are employed in flow and transport simulations. Mass-transfer across the matrix–fracture interface is implemented using the specular reflection boundary condition. Tracer partitioning coefficients can vary among the tracer compounds and in space. We use our model to match a field tracer injection test designed to determine remaining oil saturation. By analyzing time-dependent behavior of the fully-resolved, three-dimensional "fracture"–matrix geometry, we show that the industry-standard approach may consistently overestimate remaining oil saturation. For a highly heterogeneous reservoir system, relative error of the field-based remaining oil estimates may exceed 50%.

I. INTRODUCTION

Fluid tracers can illuminate flow and distribution of immiscible fluids in the subsurface. These tracers are injected to study natural or artificial processes and systems, such as aquifers[1–4], oil[5] and gas[6] reservoirs, as well as morphological and hydraulic properties of soils[7], and groundwater flows[8,9]. Tracer transport is governed by the principles of chromatography[10]. The soluble and sometimes partitioning tracers are injected into the flowing fluid phase, which carries them through the system of interest; tracer interactions with the stationary phases, e.g., rock matrix and/or immobile oil are a source of useful information at the producing wells. Usually, tracers are the easily detectable, "bright" chemical or radioactive compounds[11,12].

Motion of tracers in the subsurface is obscured by the complexity of geological formations, and establishing relations between tracer behavior and pore space geometry is still an open question. One of the main difficulties is the existence of multiple geometric length scales, formed over multiple geological time scales, and the challenges in providing detailed three-dimensional (3D) reconstructions of these geometries[13,14]. Computer simulations are a useful tool, since they capture some of the features of subsurface geometry and transport processes. In this study, we address one of the key properties of subsurface rocks — presence of fractures or high permeability flow channels — and perform transport simulations in 3D "fractured" rock geometries. Our simulation approach incorporates fractures and rock matrix, as well as mass transfer between them. Thus, we create effectively a dual-porosity system addressed previously by Warren and Root[15], Kazemi et. al.[16], Małoszewski and Zuber[17], de Smedt and Wierenga[18], Geiger et. al.[19]. Our "fractures" can also idealize the complex three dimensional networks of very high rock permeability, such as the "super-k" channels in limestones[20].

"Fractures" are generated using the polyhedral Voronoi tessellations. Once a fractured geometry is obtained, it is challenging to mesh it well enough to make the high-fidelity flow and transport calculations possible[21–24]. Often meshing is a major bottleneck in the overall model performance. We use uniform cubic mesh and develop a parallel discretization software to enable high-resolution, fast meshing of the generated 3D fractured geometries.

After flow geometry is generated and meshed, we perform transport simulations using lattice-Boltzmann (LBM) and random-walk particle-tracking methods (RWPT). Both methods are an alternative to the classical solution of advection–diffusion equation. LBM and RWPT are well-suited for high-performance computing systems, and render accurate results using regular meshes[25]. RWPT does not introduce numerical dispersion, but is subject to statistical noise, which makes generation of smooth concentration profiles computationally demanding.

In this study, we validate in detail mass transfer between the "fractures" and matrix. After establishing the simulation framework, we apply it to a practical problem of estimating remaining oil saturation from tracer injection tests.

In summary, the presented simulation approach includes the following key steps:

- Generate seed points and perform a Voronoi tessellation on them, shrinking each Voronoi volume towards its center to release void space for "fractures";

[a]Electronic mail: siarhei.khirevich@kaust.edu.sa

- Initialize a uniform cubic mesh using the Voronoi geometry obtained at the previous step;
- Simulate Stokes flow in the "fractures" with the lattice-Boltzmann method using the initialized mesh;
- Simulate the advection–diffusion transport and mass transfer of tracers with the random-walk particle-tracking method using the mesh and flow field from the previous steps.

II. METHODS

A. Geometry generation and discretization

Our approach to obtaining three-dimensional fractured geometry is based on the geometrical Voronoi tessellation[26]. Polyhedral footprint — an inherent property of our approach — appears in various formation-related processes including drying, cooling, or recrystallization[27], which makes it an attractive way of creation of the fractured geometries. In addition, polygonal/polyhedral features exist in various natural formations, as shown in Fig. 1. The high permeability flow channels in limestones, the burrowing channels left by ancient shrimp, *Thalassinoides*, are shown in Fig. 2.

In this section, we describe the key geometry algorithm steps and provide program implementation details.

1. "Fracture" generation from Voronoi tessellation

Voronoi tessellation has been used to generate fragments in, for example, computer graphics[33], or to imitate damage of brittle rocks[34,35]. In a three-dimensional Voronoi tessellation, one distributes a set of seed points in the target volume V, and then splits V into the arbitrarily-shaped polyhedra that fill V without gaps and overlaps. Polyhedral shape of each Voronoi element originates from the geometric rule according to which V is split. Each Voronoi element contains the region of space closer to a given seed point than to any of its neighbors. The two upper panels of Fig. 3 show consecutive steps of the Voronoi tessellation seeded by a choice of points in the volume of interest. Statistical properties of regular and random (Poisson) Voronoi tessellations are discussed in detail by Kumar et. al.[36] and Lucarini[37].

We obtain a fractured geometry after the following steps (see the schematic two-dimensional illustration in the bottom panel of Fig. 3):

- Distribute seed points in the volume of interest; the distribution pattern can be arbitrary, ordered or random. This pattern plays a pivotal role in the shape of the generated "rock fragments" or "fractures" between them;
- Use Qhull software[38] to perform Voronoi tessellation on the seed points and obtain a polyhedral tiling of space. The polyhedra share each of their faces with one of the neighbors, and therefore the entire tessellation can also be described as a set of polyhedron faces that are polygons (planar objects) oriented in three dimensions. In our simulations the
  Cartesian directions;
- Consider each polygon as a basis of a prism or, in other words, a polyhedron with two parallel congruent bases. Introducing normal $\vec{n}$ to the plane of a polygon, the two bases of the prism are obtained by two translations of the original polygon along $\vec{n}$, each in opposite direction but equal distance; the interior of such a prism becomes a planar "fracture." This step can also be understood as shrinking each Voronoi polyhedron towards its center to create voids for fractures.

At this stage, the resulting geometry can be seen as a set of prisms that is mapped onto a uniform cubic mesh by initializing each mesh voxel as `fracture` or `matrix`, according to the location of the voxel's center, inside or outside of the closest prism. The output cubic mesh is then used as input for transport simulations.

We note that in standard Discrete Fracture Network (DFN) approach fractures are modeled as the $(D-1)$-dimensional objects in a D-dimensional space: for example, polygons in 3D. The $(D-1)$-dimensional fractures can be discretized using uniform cubic mesh[39]. Here we use a different approach, we embed the D-dimensional "fractures" of finite width in a D-dimensional space. The additional computational effort is offset by the use of supercomputing facilities and efficient program implementation in the low-level programming languages. On the other hand, we avoid many intrinsic difficulties with meshing or flow and transport simulations, which arise from the need to connect the $(D-1)$-dimensional geometries and fields with their D-dimensional counterparts.

2. Program implementation

Voronoi tessellation is accomplished with Qhull software[38]. Qhull is called by the MATLAB routines to perform pre- and post-processing steps for i) distribution of seed points in the volume of interest, and ii) storing the data on Voronoi faces into a file. This file is then used by discretization software at the final step.

Our focus is the development of high-performance discretization code that maps arbitrary fractured volume onto a discrete mesh in three dimensions. The code was written in C language and parallelized.

The discretization code creates a uniform cubic mesh and then uses Voronoi faces together with a prescribed "fracture" width to introduce prisms ("fractures") and to mark each mesh voxel according to its location relative

A: Bitumen-infilled fractures within the Devonian Woodford Shale, Oklahoma, USA

B: Kometan Formation, Zagros Basin, Kurdistan, North East Iraq

C: Tundra polygons, Deadhorse (Prudhoe Bay), Alaska, USA

FIG. 1. Occurrence of polygonal/polyhedral shapes in natural formations. A: Reprinted with permission from the study of An *et al.*[28] (J. Pet. Sci. Eng. 157, 273 (2017)), Copyright 2017 Elsevier. B: Reproduced with permission from PhD thesis of F. Rashid[29]. Copyright 2015 F. Rashid. C: Pictures reproduced from `www.vickibeaver.com` with permission of V. Beaver[30]. Copyright 2011 V. Beaver

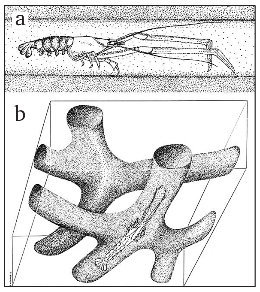

FIG. 2. Left: A very large *Thalassinoides suevicus* burrow network in the Zohar Formation (Middle Jurassic, Callovian) of Makhtesh Qatan in the Negev of southern Israel. Reproduced with permission of M. Wilson[31]. Copyright 2011 M. Wilson. Right: Reconstruction of *Mecochirus rapax* in a Cretaceous *Thalassinoides.* a) In its burrowing life mode; b) Predominantly horizontal *Thalassinoides suevicus* burrow systems showing two successive event levels, with *Mecochirus* in life position. Reproduced with permission from the study of Carvalho, Viegas, and Cachao[32] (Palaios 22, 107 (2007)). Copyright 2007 Society for Sedimentary Geology.

to the prism's interior. The code iterates over Voronoi polygons in the outer loop, constructs a prism, finds a rectangular mesh subdomain enclosing the prism, and in the inner loop processes all the voxels within this subdomain, see the top panel of Fig. 4. Each voxel center v is analyzed in three steps:

- The distance d to the closest Voronoi face plane is checked, see Fig. 4b,
- If d is less than the half-width of the prism, then projection v' of the voxel center onto Voronoi face plane is calculated (Fig. 4b);
- The code calculates the angle α_{tot} between each pair of polygon points and the projection v' (Fig. 4c), and if $\alpha_{tot} = 2\pi$ then the voxel center v is located within the "fracture" and marked accordingly.

The code is parallelized according to the Message Passing Interface (MPI) standard to enable use of computational facilities with distributed memory. We use a simple one-dimensional decomposition that splits the simulation domain — a uniform cubic mesh — into "slices", one slice per MPI process. Additionally, each MPI process handles the Voronoi face prisms intersecting a given slice (Fig. 5a).

3. Performance of discretization code

For performance tests we use a random distribution of seed points for the Voronoi tessellation. The points are distributed in the boxes of $20 \times 5 \times 5l^3$, $200 \times 5 \times 5l^3$, and $2500 \times 5 \times 5l^3$, where l is an arbitrary length unit. The density of seed points is one point per l^3 (i.e., the

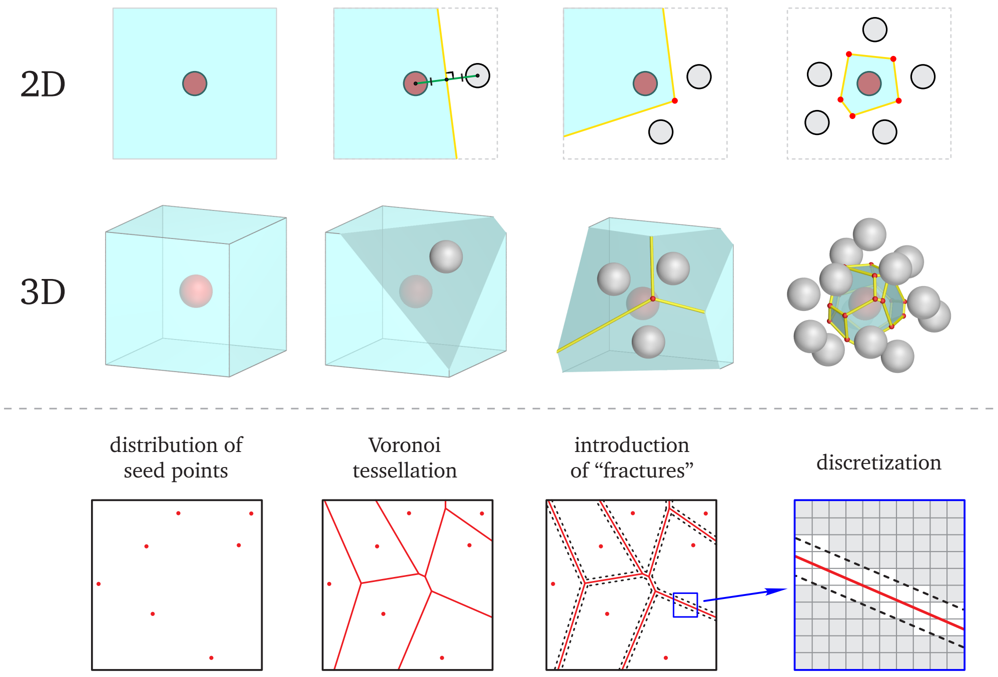

FIG. 3. Top and middle panels show construction of the Voronoi tessellations as a function of the number of seed points in the volume of interest. The bottom panel illustrates key steps of fracture generation and discretization that opens the resulting geometry to flow and transport simulations.

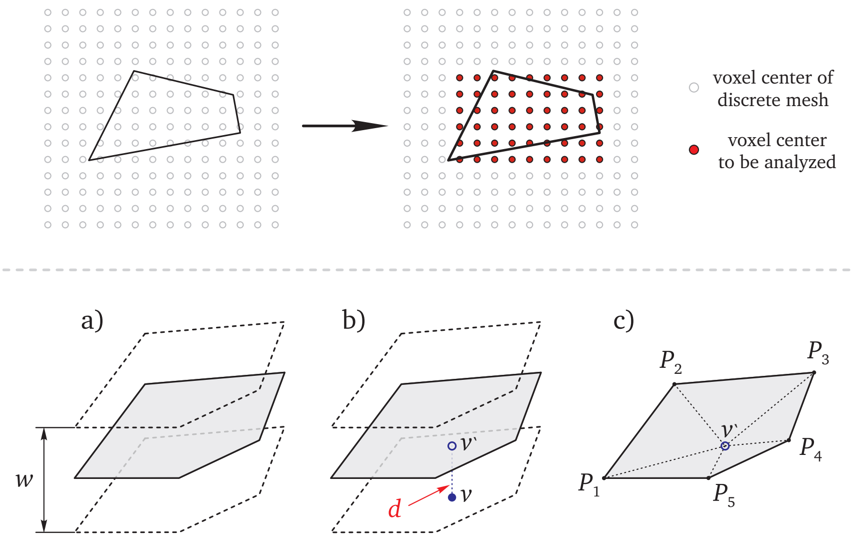

FIG. 4. Top: extraction of mesh voxels comprising a rectangular domain around a prism for further processing. Bottom: a) a "fracture" of width w formed by two images of input Voronoi face. b) Calculation of the distance d between the mesh voxel v and the Voronoi face plane, v' is the projection of v onto Voronoi face if $d < w/2$. c) Calculation of the angles between the projection v' and all Voronoi face points $P_{1\ldots5}$ (c).

first geometry has 500 seed points). Replication of the generated points to impose periodic boundary conditions along all Cartesian directions, results in about 10^5, 10^6, and 10^7 "fractures" per each generated geometry. Hereafter each geometry is discretized with the resolution of 256 mesh voxels per l, resulting in 10^{10}, 10^{11}, and 10^{12} of voxels, respectively. For these computational tasks we vary the number of employed CPU cores. The left panel of Fig. 5b shows scaling of computational performance together with absolute computational time (in seconds). In an additional performance test, we vary discretization resolution (128, 256, 512 voxels per l) for the second geometry with 10^6 fractures. The corresponding numbers of mesh voxels are about 10^{10}, 10^{11}, and 10^{12}.

Performance tests are done on the Shaheen II supercomputer (KAUST, Saudi Arabia) equipped with the Intel Haswell E5-2698v3 CPUs. The results show acceptable scaling up to several thousand CPU cores.

To our knowledge, the commonly used discrete fracture network approach to generate fractures currently deals with at most 10^4 fractures and 10^7 triangular mesh elements[22,40]. The methodology presented here is capable to generate and mesh a much larger number of "fractures." The discretization procedure avoids difficulty with the intersections of multiple fractures at one point, which is a challenge for meshing (triangulation) of DFN geometries. Namely, each fracture is processed independently, and the mesh voxels at fracture intersections can be reinitialized as many times as needed. Our code is not limited to Voronoi tessellations and could be modified to mesh various non-polygon planar objects. The large number of the generated "fractures" enables, for example, introduction of several length scales, and the high discretization resolution allows adding fine features to each fracture, such as surface roughness.

B. Flow simulations: the lattice-Boltzmann method

To simulate Stokes flow in the generated "fracture" geometries, we employ the Lattice-Boltzmann method (LBM). LBM is a powerful approach for pore-scale simulations, which gained increased attention over recent decades[41,42]. LBM does not solve the (Navier–)Stokes equations directly, but recovers flow behavior through simulation of evolution of a *lattice gas*. The lattice gas is represented by a distribution function of its local "fluxes" over lattice nodes. Each flux can propagate along a discrete set of lattice links ($j = 0, \dots, 18$ in this study) connecting each lattice node to its neighbors ($j = 1, \dots, 18$) as well as to itself ($j = 0$). Evolution with time t of a flux along the j-th lattice link at the lattice node $\mathbf{r}$ can be described by its distribution function $f_j(\mathbf{r}, t)$:

$$\tilde{f}_j(\mathbf{r}, t) = f_j(\mathbf{r}, t) + \Omega(f_j), \tag{1a}$$

$$f_j(\mathbf{r} + \mathbf{e}_j \delta t, t + \delta t) = \tilde{f}_j(\mathbf{r}, t), \tag{1b}$$

where δt denotes the simulation time step, $\mathbf{e}_j$ is the lattice vector along the j-th link, and Ω is the collision operator. According to Eqs. (1), in each iteration LBM i) performs the collision of fluxes at each lattice node (Eq. (1a)), and ii) moves the fluxes among the lattice nodes along the lattice links (Eq. (1b)). LBM is constructed in such a way that with elapsing time the distribution function iteratively approaches its equilibrium state. The simulation is halted when the distribution function stops changing, or, alternatively, equilibrium is reached up to a given accuracy.

LBM implementations differ in their collision operator. One of the simplest and commonly used is the Bhatnagar–Gross–Krook (BGK) collision operator, originating from the dynamic theory of gases[43]. It contains only one parameter, the relaxation time, τ:

$$\Omega(f_j) = -\frac{f_j - f_j^{eq}}{\tau}. \tag{2}$$

The single relaxation parameter τ determines both the relaxation rate of the distribution function towards its equilibrium (f_j^{eq}), as well as the viscosity of the simulated fluid[44]. When performing LBM simulations on the micro-CT images, the bounce-back boundary condition is commonly used. This boundary condition reflects backwards the fluxes facing solid lattice nodes. Combination of the bounce-back BC and BGK model is well known for its dependency of the simulated permeability on τ, e.g. Ginzburg and d'Humières [45].

Reformulation of the LBM matrix-collision model led to the development of the multiple-relaxation-times collision operator (MRT)[46,47]. In MRT, a collision occurs in the space of hydrodynamic and kinetic moments (like density, momentum, energy, energy flux), with the transformation of the distribution function to the momentum space and vice versa. The operator itself can be written as

$$\Omega(f_j) = -(\mathtt{M}^{-1})_{jk} \hat{\mathtt{S}}_{ki} (m_i(\mathbf{r}, t) - m_i^{eq}(\mathbf{r}, t)), \tag{3}$$

where the components m_i represent the moment space to which the distribution function f_j is converted via the transformation matrix $\mathtt{M}$. $\hat{\mathtt{S}}$ denotes the diagonal matrix, whose elements, $\hat{s}_i$, are the reciprocal values of the relaxation times τ_i associated with the moments m_i, $i = 0, \dots, 18$. The multiple relaxation times in (3) provide the capability of adjusting each parameter τ_i individually. This capability poses the yet-unanswered questions about the particular choices of those parameters. Adjustment of multiple MRT relaxation rates is useful, for example, in full Navier–Stokes simulations, where it improves stability in high Reynolds number flows. It was suggested that for the Stokes-only simulations a reduced version of MRT operator, or the two-relaxation-times operator (TRT), is a better choice[48,49]. In TRT, all MRT rates are combined into two groups, and each group is assigned a single relaxation rate, τ_ν and τ_f. In the simulations, one adjustable rate (τ_ν) controls viscosity of the simulated fluid while the second one (τ_f) appears free-tunable, but in fact controls the resulting permeability k. In particular, if the quantity $\Lambda = (\tau_\nu - 0.5)(\tau_\mathrm{f} - 0.5)$ is kept constant, k stays fixed up to machine accuracy for any geometry[50]. Therefore, the simulated permeability depends on a particular choice of Λ. In our study, we use a single Λ-value of $3/16$ for all simulations[49].

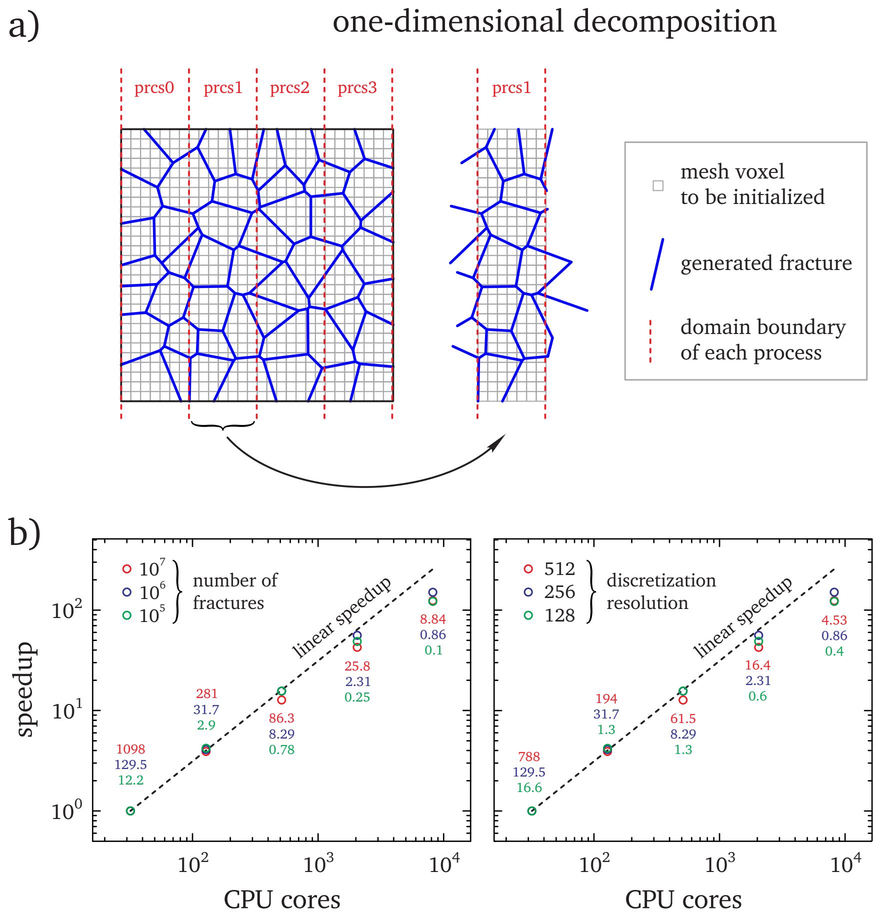

FIG. 5. a) Schematic illustration of one-dimensional domain decomposition between MPI processes: each process analyzes its own subset of mesh voxels as well as part of the Voronoi prisms (fractures) crossing the domain of that process. b) The speedup of discretization performance vs. increase of CPU cores for a random fracture distribution. Performance of 32 cores is normalized to one, and further 128, 512, 2048, 8192 cores are used for the larger CPU numbers. The left panel shows speedup for the geometries with the different numbers of fractures (left legend), discretized at 256 voxels per unit length. The right panel deals with the geometry of 10^6 fractures discretized at different resolutions shown in the right legend. The numbers near each data point denote the corresponding absolute execution time in seconds of the discretization procedure.

C. Tracer dynamics: the random-walk particle-tracking method

After obtaining a stationary flow field from the lattice-Boltzmann simulations, we use the Random-Walk Particle-Tracking approach (RWPT)[51–54] to simulate transport of tracer particles in the generated "fractured" geometries. RWPT uses a large number of infinitely small particles, and in each iteration displaces each particle with the diffusive and advective components:

$$\begin{aligned}\Delta\mathbf{r} &= \mathbf{r}(t+\Delta t) - \mathbf{r}(t) = \\ &= \mathbf{v}(\mathbf{r})\Delta t + \xi\sqrt{2D_{\text{region}}(\,\mathbf{r}(t)\,)\Delta t}\end{aligned} \tag{4}$$

where $\mathbf{r}$ is the tracer particle coordinate at time t, Δt is the time step, $\mathbf{v}$ is the velocity vector at location $\mathbf{r}$, and D_{region} is the diffusion coefficient in the *region* of the voxel whose center is closest to $\mathbf{r}$. The term ξ denotes a random vector obtained from the standard normal distribution of $N(0,1)$. Tracer concentration can be found from the number of tracers occupying a given number of voxels. The variation of Péclet number is obtained from linear scaling of velocity components using linearity of the Stokes flow solution.

Cubic mesh used in the LBM simulations provides voxels of two types, `fracture` and `matrix`. (We also refer to these voxel types as "regions".) This mesh is used for tracer simulations, too. The "fracture" voxels are assumed to contain only water, while the matrix voxels consist of water, oil, and rock, see Fig. 6a. In this study, the effective tracer concentration in a `matrix` voxel, C_{m}, is a function of oil (water) saturation, S_{o} (S_{w}), and the matrix porosity, ϕ_{m}. The oil (water) saturation is the fraction of the matrix pore volume occupied by oil (water). For example, if $\phi_{\text{m}} = 0.1$ and $S_{\text{o}} = 0.1$, oil will occupy 1% of the total matrix volume.

We consider two types of tracer particles: the "partitioning," or active tracers, and the "non-partitioning," or passive, conservative ones. The partitioning tracers are soluble in water and oil, while the non-partitioning ones are only water-soluble. Their concentrations in `matrix` voxels, C_m^p and C_m^np, can be defined as

$$C_\mathrm{m}^\mathrm{p} = \frac{n_\mathrm{m}^\mathrm{tr}}{n_\mathrm{m}^\mathrm{vox}\phi_\mathrm{m}}, \tag{5a}$$

$$C_\mathrm{m}^\mathrm{np} = \frac{n_\mathrm{m}^\mathrm{tr}}{n_\mathrm{m}^\mathrm{vox}\phi_\mathrm{m}S_\mathrm{w}} \tag{5b}$$

where n_m^tr is the number of tracers occupying $n_\mathrm{m}^\mathrm{vox}$ number of `matrix` voxels. The corresponding concentration of partitioning and non-partitioning tracers in $n_\mathrm{f}^\mathrm{vox}$ `fracture` voxels, C_f^p and C_f^np, is

$$C_\mathrm{f}^\mathrm{p} = C_\mathrm{f}^\mathrm{np} = \frac{n_\mathrm{f}^\mathrm{tr}}{n_\mathrm{f}^\mathrm{vox}}, \tag{6}$$

because we assume $S_\mathrm{w} = 1.0$ in `fracture` region. At equilibrium, the concentration of tracer particles of the same type in `fracture` and `matrix` regions is the same. For example, if $\phi_\mathrm{m} = 0.2$, $S_\mathrm{w} = 1.0$, and $n_\mathrm{f}^\mathrm{vox} = n_\mathrm{m}^\mathrm{vox}$, then there are five times fewer tracers in `matrix` region at long simulation times.

The partitioning coefficient,

$$K = C_\mathrm{o}/C_\mathrm{w}, \tag{7}$$

determines the *equilibrium* tracer concentration ratio between the immobile oil and water phases located only in `matrix` region. Here K is independent of the water phase located in `fracture` region. For partitioning tracers, variation of K changes their equilibrium concentration in `matrix` region, which also impacts their concentration in `fracture` region. Equality of concentrations in the two regions at equilibrium can be formulated using K as follows:

$$\frac{n_\mathrm{f}^\mathrm{tr}}{n_\mathrm{f}^\mathrm{vox}} = \frac{n_\mathrm{m}^\mathrm{tr}}{n_\mathrm{m}^\mathrm{vox}}\frac{1}{\phi_\mathrm{m}(S_\mathrm{w}+S_\mathrm{o}K)}, \text{ or} \tag{8a}$$

$$R = \frac{n_\mathrm{f}^\mathrm{tr}}{n_\mathrm{m}^\mathrm{tr}} = \frac{n_\mathrm{f}^\mathrm{vox}}{n_\mathrm{m}^\mathrm{vox}}\frac{1}{\phi_\mathrm{m}(S_\mathrm{w}+S_\mathrm{o}K)}. \tag{8b}$$

The last equation can be used to validate accuracy of the simulation code by comparing the simulated and analytical values of the ratio $R = n_\mathrm{f}^\mathrm{tr}/n_\mathrm{m}^\mathrm{tr}$.

Tracer particles are in constant motion, and to ensure correct model behavior, the following requirements must be satisfied:

- independence of equilibrium tracer concentration on the tracer diffusion coefficients in `fracture` and `matrix` regions;
- linear dependence of tracer concentration on the matrix porosity; and
- maintenance of the correct equilibrium tracer concentration after a change of oil (water) saturation and partitioning coefficient.

Tracer particles are not aligned with the lattice voxels and move without restrictions between voxels of the same region. When moving into a different region, each particle can change diffusion coefficient or can be reflected from the interface between these regions, see Figure 6b,c. Variation of the diffusion coefficient and the probability of tracer reflection are the basic tools to meet the requirements above. To control tracer dynamics, we use the approach discussed by Bechtold *et al.*[55] and Daneyko *et al.*[56]. Let us introduce the probability for a tracer particle to enter `matrix` from a `fracture` voxel, $P_{\mathrm{f}\to\mathrm{m}}$, and for the move in the opposite direction, $P_{\mathrm{m}\to\mathrm{f}}$. A desired ratio of equilibrium concentrations between the `matrix` and `fracture` regions can be obtained for the different values of $P_{\mathrm{f}\to\mathrm{m}}$ and $P_{\mathrm{m}\to\mathrm{f}}$, but at their fixed ratio. From the point of reducing computational demand, one of these probabilities can always be set to 1. Similar arguments are applicable to the diffusion coefficients within the `fracture` region, D_f, and the `matrix` region, D_m. We use the following rules to control motion of each tracer particle:

$$\begin{cases} P_{\mathrm{f}\to\mathrm{m}} = K'\phi_\mathrm{m}\sqrt{D_\mathrm{m}}/\sqrt{D_\mathrm{f}} \text{ and } P_{\mathrm{m}\to\mathrm{f}} = 1 \\ \qquad\qquad \text{if } K'\phi_\mathrm{m}\sqrt{D_\mathrm{m}}/\sqrt{D_\mathrm{f}} < 1, \\ P_{\mathrm{m}\to\mathrm{f}} = (K'\phi_\mathrm{m}\sqrt{D_\mathrm{m}}/\sqrt{D_\mathrm{f}})^{-1} \text{ and } P_{\mathrm{f}\to\mathrm{m}} = 1 \\ \qquad\qquad \text{otherwise.} \end{cases} \tag{9}$$

Here $K' = S_\mathrm{w} + KS_\mathrm{o}$ with K defined in (7). For $K = 0$ the model recovers behavior of non-partitioning tracer, while partitioning tracer is simulated with $K > 0$.

When a tracer particle crosses the interface between different regions, it changes its diffusion coefficient. Consider the case when particle jumps from a `fracture` to a `matrix` voxel, see Figure 6c. The original tracer jump vector $\Delta\mathbf{r}_{\mathrm{f}\to\mathrm{m}}$ is calculated using the time step Δt and the diffusion coefficient in `fracture`, D_f. $\Delta\mathbf{r}_{\mathrm{f}\to\mathrm{m}}$ can be seen as the sum of two vectors, one located in the `fracture` ($\Delta\mathbf{r}_\mathrm{f}$) and another in the `matrix` ($\Delta\mathbf{r}_\mathrm{m}$) voxels. Let us consider a non-linear time-splitting scheme: $\sqrt{\Delta t} = \sqrt{\Delta t_\mathrm{f}} + \sqrt{\Delta t_\mathrm{m}}$, where Δt_f and Δt_m are the times spent by tracer in `fracture` and `matrix` regions, respectively. Using this time-splitting scheme and the Einstein diffusion relationship, we apply the following correction to take into account the change in diffusion coefficient for the jump vector $\Delta\mathbf{r}_{\mathrm{f}\to\mathrm{m}}$:

$$\Delta\mathbf{r}_{\mathrm{f}\to\mathrm{m}}{}^* = \Delta\mathbf{r}_\mathrm{f} + \Delta\mathbf{r}_\mathrm{m}\sqrt{D_\mathrm{m}/D_\mathrm{f}}, \tag{10a}$$

$$\Delta\mathbf{r}_{\mathrm{m}\to\mathrm{f}}{}^* = \Delta\mathbf{r}_\mathrm{m} + \Delta\mathbf{r}_\mathrm{f}\sqrt{D_\mathrm{f}/D_\mathrm{m}}, \tag{10b}$$

where the last equation defines a jump in the opposite direction, from `matrix` to `fracture` voxels.

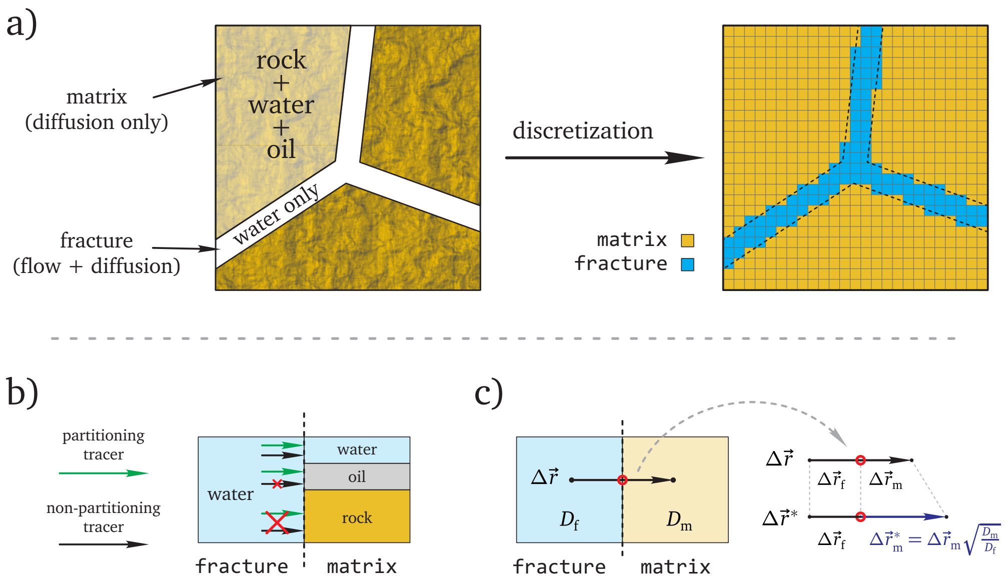

FIG. 6. a) Schematic presentation of the combination of pore-scale and effective medium approaches used in this work. Tracer simulations are performed in the discretized fractured geometry with voxels of `fracture` and `matrix` types. The flow field in `fracture` voxels is obtained from LBM simulations. b) Implementation of the effective medium approach: for a jump from `fracture` voxel, a partitioning tracer can enter `matrix` voxel through water or oil phases, while a non-partitioning one through water only; rock reflects both tracers. Presence of rock, oil, and water in `matrix` region is not resolved explicitly and is simulated using a probability of reflection. c) Correction of a tracer jump, when the tracer enters a voxel with a different diffusion coefficient.

1. Boundary conditions for RWPT

A special attention is given to the choice of the boundary condition (BC) applied to the tracer particle crossing the interface between the `matrix` and `fracture` regions. Particle jump can be denied with some probability, and there are different ways of implementing this rule in a simulation code. Szymczak and Ladd[57] summarized several approaches, such as "rejection", "interruption", "multiple rejection", and "specular reflection". Rejection cancels the jump if it is not allowed, and a tracer particle does not change its coordinate. Multiple rejection regenerates a stochastic part of a particle jump (the last term in (4)), until the particle ends up at the allowed location. Interruption BC i) splits the original jump into two vectors, prior and after the point of intersection with the interface, and ii) moves the tracer particle along the first vector and then performs a new random jump with the magnitude of the second vector. Specular reflection is similar to interruption, but the second vector is oriented in the direction of mirror reflection of the original jump. As shown by Szymczak and Ladd[57], only specular reflection preserves correct concentration profile close to the reflective wall. (Here "close" means the distance of a single tracer jump.) In the case of a fully-reflective interface, the distortion of concentration profile due to the employed BC can be reduced by decreasing the time step (smaller tracer jump). However, this is not the case when the interface is semi-reflective, as we demonstrate below.

Some aspects of our simulation approach, such as accuracy of the lattice-Boltzmann simulations[25] and hydrodynamic dispersion were addressed previously (pp. 40–42 in[58]). Here, we assess only accuracy of the cross-interface mass transfer using several tests. The first test is a comparison of simulation results with an analytical solution by Scott, Tung, and Drickamer[59]. This solution provides a one-dimensional, time-dependent distribution of concentration between two cells of equal length separated by an interface. The domain containing these two cells is confined by the impermeable walls from left and right. Initially, the left cell has a concentration $C_0 = 1$ while the right cell is empty; each cell can have a different diffusion coefficient. Tracers are allowed to diffuse through the interface with a discontinuity in the diffusion coefficient. Partitioning (or distribution) coefficient K in this study equals $K = C_{\text{right}}/C_{\text{left}}$, while Scott et al. used its reciprocal value. After solving the classical diffusion equation, the authors provide a solution for the concentration in left and right cells in terms of series, cf. eqs. (23) and (24) in[59] requiring roots of their eq. (25). We have solved (25) in[59] up to 1000 roots with MATLAB using the CHEBFUN package[60]. Please note that our correction to (23) replaces $C_0 m$ with C_0.

Fig. 7a compares tracer simulation results with the solution of Scott et al. for three time intervals: 1, 10, and 10000 tracer particle jumps or iterations. We vary partitioning coefficient ($K = 0.2$ and 5), and use non-equal diffusion coefficients in `fracture` and `matrix` regions, $D_m = 0.25 D_f$. While our typical simulation takes about 10^9 iterations, analysis of the shorter time scales provides better insight into the employed numerical approach. In the case of specular reflection, Fig. 7a shows the almost identical results for 10000 time steps, while there is some difference for one and a few time steps. This difference can be explained by the assumption of instantaneous concentration equilibrium at the interface used by Scott et al. via the prescribed interface condition, see the discussion of eqs. (7)–(9) in Scott, Tung, and Drickamer [59]. In fact, it takes some time to achieve equilibrium at the interface. We conclude that in this regard our tracer model imitates reality better than the solution of Scott, Tung, and Drickamer [59]. The multiple rejection boundary condition fails to recover correct concentration profile in all considered cases.

Fig. 7b addresses the importance of selection of boundary condition in tracer simulations, when a semi-reflective interface presents. In the case of a fully-reflective wall, the multiple rejection BC may distort tracer concentration near the wall[57]. For a semi-reflective wall, this distortion changes flux of tracers between two phases, which results in an incorrect distribution of equilibrium concentrations. To demonstrate a possible impact of a concentration distortion in higher dimensions, we have performed a simulation with the input parameters providing uniform tracer concentration for the whole simulation domain. Domain size is $2 \times 2 \times 2$ voxels, and periodic boundary conditions are applied along all Cartesian directions. Panel b in Fig. 7 reveals that, depending on simulation parameters, multiple-rejection BC may introduce a significant concentration maldistribution, while specular reflection recovers the correct behavior.

As the last validation test, we have quantified relative error in the concentration ratios for several geometrical configurations: layered, "chess", and random (top of Fig. 7c), all with periodic boundary conditions. The error is calculated according to eq. (8b), introducing R_{analyt} as the reference equilibrium concentration ratio between `fracture` and `matrix` regions, and R_{sim} as its simulated counterpart. The layered configuration can be assumed as the reference flat interface, while the chess configuration adds corners that may degrade accuracy of specular reflection due to the multiple reflections of a single jump of a tracer particle. We note that our numerical implementation allows each particle jump to have an arbitrary number of reflections. Random configuration is obtained from a 32x32x32 cube filled with 50% of `fracture` and 50% of `matrix` voxels. The model parameters are as follows: $D_{\text{m}}/D_{\text{f}} = 1$ or 0.1, $K = 5$, $S_{\text{o}} = 1.0$, $\phi_{\text{m}} = 0.1$. A variation of the diffusion coefficient ratio should not change equilibrium distribution; however, it is related to the change in reflection probability, and therefore may change the numerical results. Additionally, we vary tracer time step (Δt) to address its possible impact on the reduction of numerical error. We use the default value of Δt, resulting in root-mean-squared-displacement of ~ 0.167 of voxel edge, and its value reduced five times, i. e. the time step of $0.2\Delta t$.

For all geometrical configurations and diffusion coefficient ratios, the specular reflection BC outperformed multiple rejection (blue/red symbols vs. green/black ones). Specular reflection creates the largest error in random configuration and the highest difference between the diffusion coefficients in `matrix` and `fracture` regions. Reducing time step (circles vs. dots) improves accuracy of specular reflection, mostly in random configuration, but does not allow to go significantly below 1%. For the multiple rejection BC, reduction of the time step does not reduce the error, because this BC distorts tracer concentration profile near the interface on a distance of a single jump. We note that for some sets of parameters, multiple rejection can produce smaller or larger error relative to the one shown in Fig. 7c. For a random configuration and specular reflection BC, the case of $D_{\text{m}}/D_{\text{f}} = 0.1$ results in error higher than that in $D_{\text{m}}/D_{\text{f}} = 1$. This error can be reduced using smaller time steps. We attribute the error increase to numerical errors in floating-point arithmetics.

In the tests described above tracer particles diffuse without advection. Adding flow may impact accuracy of the results, and this is discussed later.

III. RESULTS AND DISCUSSION

In petroleum industry, remaining oil saturation is estimated from the production histories of tracer compounds mixed in with a slug of water injected into a partially depleted reservoir[61]. In the particular case of carbonate reservoirs in the Middle East, tracer flow geometry is dominated by the super-k channels and/or fractures. Consistently with the notation in this work, we will call both of these reservoir rock features "fractures." The rock matrix has a much lower permeability and the fluids there are almost or truly immobile. We assume that the immobile reservoir fluids filling the rock matrix are water and oil. Two types of tracers are injected, partitioning and non-partitioning, see subsection II C, eqs. (7),(9). The flow of injected water carries tracers through the "fractures" in the reservoir, and the tracers can diffuse in and out of the immobile reservoir fluids. Partitioning tracers spend more time in immobile fluids due to the finite probabilities of i) diffusion in both water and oil, and ii) entering matrix through the water and oil phases. The non-partitioning tracers are confined only to the water phase.

After passing through the reservoir, the histories of concentrations of each tracer in produced water can be estimated from a sampling observation well or producer. The remaining oil saturation can then be estimated from the following patented empirical equation[62]:

$$S_{\text{o}}^{\text{est}} = \frac{T_{\text{p}} - T_{\text{np}}}{T_{\text{p}} + T_{\text{np}}(K-1)}, \tag{11}$$

where T_{p} and T_{np} are the peak (maximum concentration) locations for the exit times of partitioning and non-

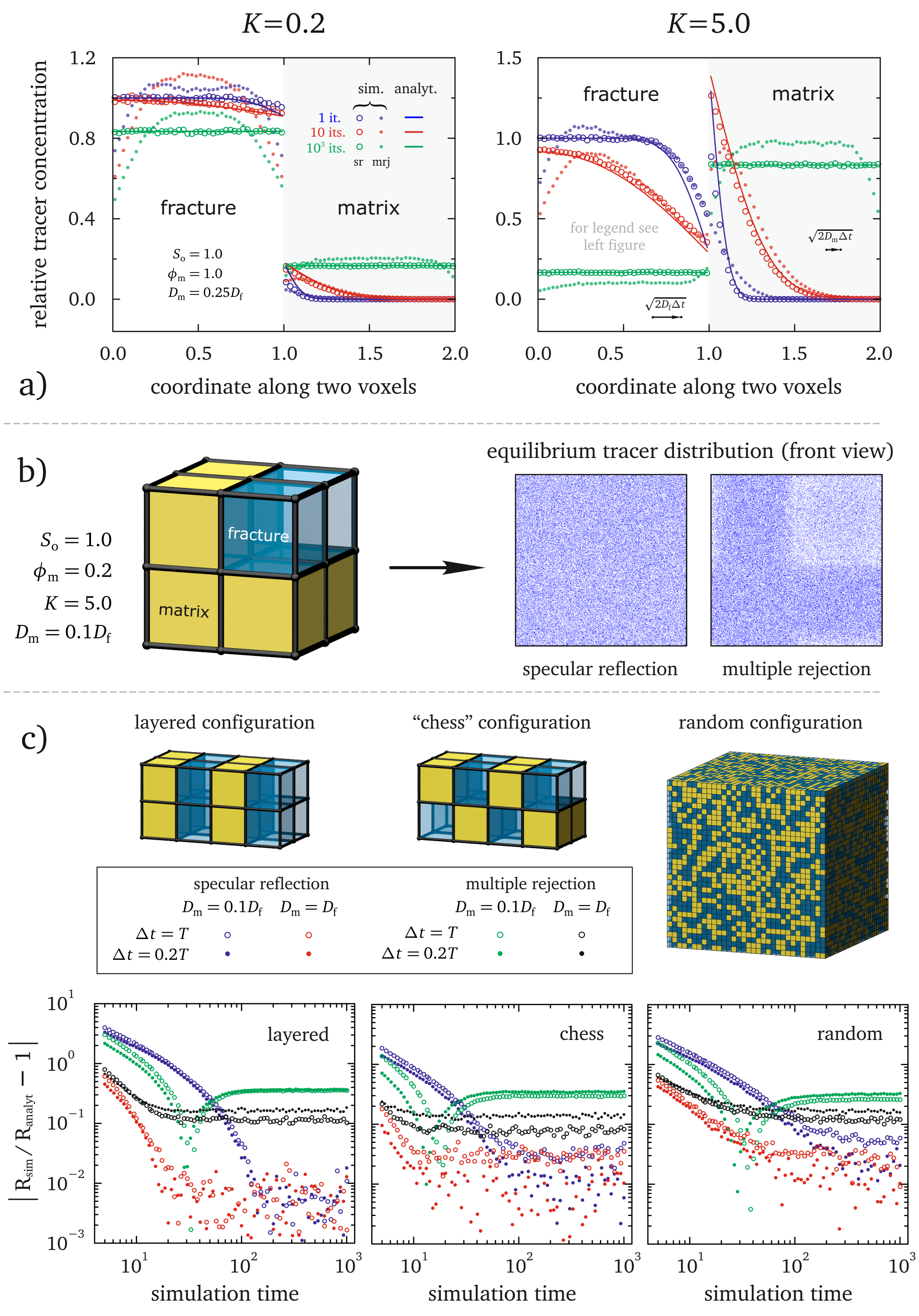

FIG. 7. a) Comparison of the numerical distribution of tracer concentration with the one-dimensional analytical solution of Scott, Tung, and Drickamer[59]. The simulations are performed in two voxels of different regions, confined by impermeable walls from left and right. Results are shown for three times. In simulations, two boundary conditions are used: specular reflection (circles) and multiple rejection (dots). Black arrows indicate the root-mean-squared-displacement of a tracer in each region. b) Illustration of equilibrium tracer maldistribution caused by the boundary condition unsuitable for mass transfer across an interface. For the chosen parameters, correct tracer distribution is uniform across the simulation domain. c) Relative error in the equilibrium concentration ratio R (see eq. (8b)) vs. simulation time (in RWPT time steps) for different geometrical configurations. The simulation parameters are: $D_m/D_f = 1$ or 0.1, $K = 5$, $S_o = 1.0$, $\phi_m = 0.1$.

partitioning tracers, respectively. We note that one *could* use the entire tracer concentration histories to estimate oil saturation, but in reality recording tails of those histories is difficult due to the long tracer retention times. Equation (11) provides the fraction of oil in the total liquid phase accessible to tracers meaning that the reference oil saturation can be defined as follows:

$$S_\mathrm{o}^{\mathrm{est},*} = \frac{\text{oil volume}}{\text{oil + total water volume}} = \\ = \frac{S_\mathrm{o}\phi_\mathrm{m}(1-\Phi_\mathrm{f})}{\Phi_\mathrm{f}+\phi_\mathrm{m}(1-\Phi_\mathrm{f})}. \tag{12}$$

Here Φ_f denotes the fracture porosity or the ratio of the number of `fracture` voxels and their total amount. We assume that oil occupies only `matrix` while water is located in `matrix` and `fracture` regions.

To analyze behavior of the presented model, we approximate the tracer tests described above. Using our Voronoi tessellation approach, we generate two sets of fractured geometries, originating from cubic and random distribution of $3 \times 3 \times 256 = 2304$ seed points in Fig. 8. Both geometries are periodic in all directions. For the cubic geometry, edge of each matrix block is assumed to have the length of 0.5m before the introduction of fractures. Random geometry has the same number of seed points, but with a different spatial arrangement. The fracture porosity Φ_f is chosen close to 4.6% in both geometries. With the discretization resolution of 256 voxels per 0.5 m, each fracture has the aperture of 5.6 voxels in random geometry or 4.0 in cubic one. Fracture porosity of 4.6% can be seen as an average value between fractured formations (below 1%) and geometries with super-k channels (above 10%)[63,64].

The matrix porosity is set to $\phi_\mathrm{m} = 0.2$ while the saturations of fluids filling each matrix block are $S_\mathrm{o} = 0.2$ for oil and $S_\mathrm{w} = 1 - S_\mathrm{o} = 0.8$ for water. According to equation (12), such a choice of simulation parameters results in $S_\mathrm{o}^{\mathrm{est},*} \approx 0.161$.

Tracer diffusion coefficients are equal in the `fracture` and `matrix` regions ($D_\mathrm{f} = D_\mathrm{m}$). Tracer simulations are performed with different number of tracer particles, ranging from $2 \cdot 10^4$ to 10^6, and for different simulation times, between 10^8 and 10^{10} iterations. These simulation parameters are needed to i) reproduce smooth estimated oil saturation curves at short time scales, and ii) achieve longer elapsed physical times within the computational time limits imposed by the supercomputing laboratory.

For the cubic "fracture" geometry, tracers are injected in the plane indicated by the red dashed lines in Fig. 8. For the random geometry, tracers are distributed uniformly throughout the `fracture` region. A uniform initial distribution of tracers is equivalent to averaging many simulation runs, each with a delta-pulse injection at a different location within the geometry. Such an approach is possible because for each tracer not only absolute coordinate, but also displacement from its origin can be tracked.

The next parameter to be chosen for the simulations is Péclet number that quantifies dominance of advective transport over the diffusive one. We define Péclet number as $\mathrm{Pe} = lv/D_\mathrm{f}$, where l is the characteristic length, v is the interstitial flow velocity, and D_f is the tracer diffusion coefficient in `fracture` region. l is taken as $(V_\mathrm{avg})^{1/3}$ where V_avg is the average volume of the rock matrix block before introducing fractures (or average Voronoi volume after the tessellation). Flow rates are chosen so that $\mathrm{Pe} = 20$, $\mathrm{Pe} = 1000$, $\mathrm{Pe} = 10000$, and $\mathrm{Pe} = 40000$. As will be shown later, values of oil saturation determined for low Péclet numbers, such as $Pe \approx 20$, and for small interwell distances demonstrate behavior quantitatively different from that for $\mathrm{Pe} = 1000$ or higher.

In our analysis, we strive to: a) perform simulations in the diffusion-dominated regime ($\mathrm{Pe} = 20$) and b) achieve realistic exit times for the distributions of tracers. For orientation, we consider the experimental field data from Sanni *et al.*[5]. The authors reported the distance of ~ 610 m between tracer injection and detection (injector–producer distance). Their data suggest that the maxima of produced tracer concentrations are at ~ 250, ~ 350, and ~ 450 days for the partitioning coefficients of $K = 0$, 2, and 4, respectively. They do not provide explicitly information about the flow rates used in the test. In our simulation setup, similar exit times for the maxima of concentration curves are obtained for $\mathrm{Pe} = 40000$. This translates into the interstitial flow rate of water in the "fractures" of $8{\cdot}10^{-5}$ m/s (or 6.91 m/d) assuming $l = 0.5$ m and $D_\mathrm{f} = 10^{-9}$ m^2/s. The corresponding Darcy or superficial velocity is then $8 \cdot 10^{-5} \times 0.046 = 3.68 \cdot 10^{-6}$ m/s or 0.32 m/d.

The left panels in Fig. 9 show the produced (exit) concentration histories of the non-partitioning and partitioning tracers for various flow rates in the two flow geometries. The area under each curve is normalized to one, even though not all of the histories are fully recorded. In other words, only the exit times, but not absolute values of the distribution functions should be analyzed. Each distribution curve is obtained from six histograms with a slightly varied number of bins. Our goal here is to consider statistical variation of the obtained results. To determine the timing of maximum concentration, 30% of the highest points in each histogram are fit with a 4-th order polynomial, and the polynomial maxima determine $S_\mathrm{o}^\mathrm{est}$. Six values of $S_\mathrm{o}^\mathrm{est}$ allow to find their mean and calculate the 95% confidence interval; both are shown in Fig. 9.

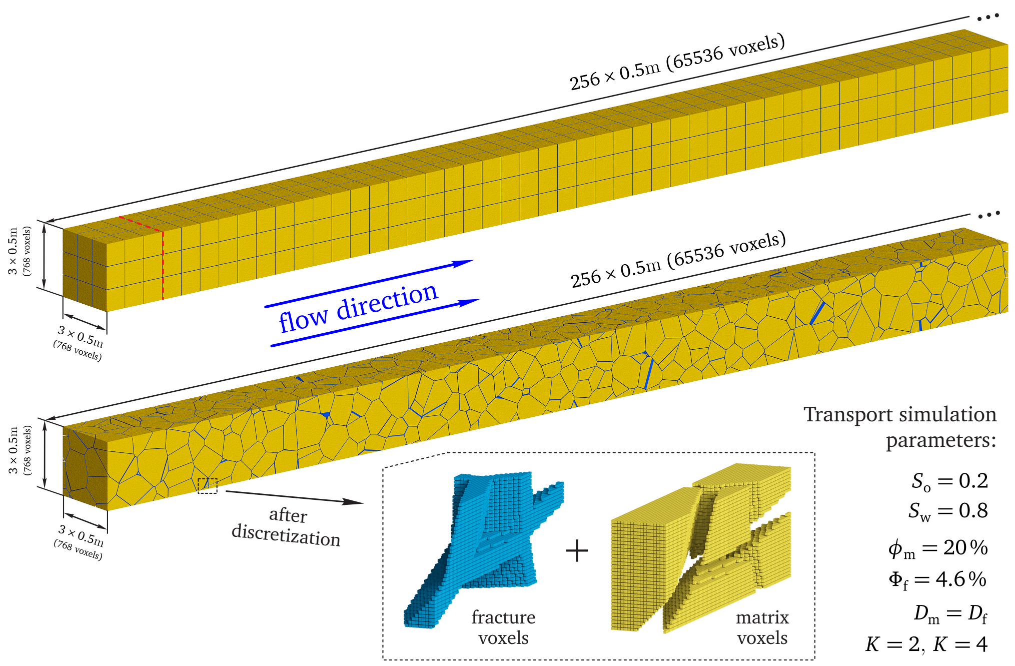

FIG. 8. Two geometries used in transport simulations. Each geometry includes $3 \times 3 \times 256 = 2304$ rock matrix blocks. For the cubic geometry shown are the first 48 (out of 256) rock blocks in the longitudinal direction; the same length is depicted for the random geometry. Both geometries have periodic boundary conditions. S_o (S_w) is oil (water) saturation in each rock matrix block; ϕ_m is its porosity. Φ_f is the fracture porosity. The diffusion coefficient of a tracer is identical inside the fractures and matrix. The inset shows a small subvolume of the random geometry discretized at the resolution of 256 voxels per 0.5 m. The red dashed line shows the injection location for tracers in the cubic geometry.

The right panels in Fig. 9 present estimated oil saturation. Initially, the estimated oil saturation is zero, because the maxima locations for each tracer coincide. As time elapses, the distance between the maxima starts to increase. Some tracers are "trapped" by the matrix, while others travel with flow. Because of diffusion, the former leave the matrix and the latter enter it. After sufficient time, tracers explore both matrix and the high-velocity "fractures," and the situation starts to repeat itself. This time corresponds to the traveled distance L, after which $S_\mathrm{o}^\mathrm{est}$ demonstrates the close-to-asymptotic behavior. As Fig 9 shows, L can vary from tens of meters for $\mathrm{Pe} = 20$ to several kilometers for the realistic flow rates obtained with $\mathrm{Pe} = 40000$. This length is similar for the cubic and random geometries, and depends on the size of the matrix blocks rather than their shapes.

Starting from $\mathrm{Pe} = 1000$, the estimated oil saturation curve demonstrates a maximum near $S_\mathrm{o,max}^\mathrm{est} \approx 0.25$ before descent to its close-to-asymptotic behavior. The value of $S_\mathrm{o,max}^\mathrm{est}$ remains approximately constant with further increase of flow rate for both geometries and chosen partitioning coefficients. This means that for the chosen simulation parameters, the maximum of $S_\mathrm{o}^\mathrm{est}$ mostly develops well before realistic flow rates are achieved. The formation of such a maximum can be seen as a source of overestimation of $S_\mathrm{o}^\mathrm{est}$.

The close-to-asymptotic value of oil saturation $S_\mathrm{o}^\mathrm{est}$ estimated by our model is identical to $S_\mathrm{o}^{\mathrm{est},*} \approx 0.161$ for the infinite injector–producer distance at lower flow rates ($\mathrm{Pe} = 20$ and $\mathrm{Pe} = 1000$), Fig. 9, right. With further increase of the flow rate, the random geometry starts to overestimate $S_\mathrm{o}^{\mathrm{est},*}$ while $S_\mathrm{o}^\mathrm{est}$ in cubic geometry stays approximately constant. We attribute this overestimation to the increasing numerical error, which is discussed below. At high flow rates, $\mathrm{Pe} \geq 10000$, the reference value of remaining oil saturation can be overestimated by up to 50–70%, depending on the injector–producer distance. For field-scale flow rate of $\mathrm{Pe} = 40000$ the distance needed to observe the close-to-asymptotic behavior is of the order of a few kilometers, which may exceed well spacing in a typical reservoir ($\approx$ 1 km). If this is the case, the existence of a maximum in $S_\mathrm{o}^\mathrm{est}$ curve may dominate field data. We remind that in this study, the diffusion coefficients of tracers in the `fracture` and `matrix` regions are assumed to be equal to reduce numerical error (Fig. 7c). In reality, the diffusion coefficient in rock matrix can be an order of magnitude smaller than in bulk fluid. This will additionally increase the equilibration time and, therefore, distance required for the close-to-asymptotic behavior in estimated oil saturation.

In tracer injection tests designed to estimate remaining oil saturation it is common practice to use tracers with several partitioning coefficients K[5,65]. This is done to improve reliability of the experimentally obtained values of $S_\mathrm{o}^\mathrm{est}$. As our results demonstrate, variation of K within a practical range ($K \approx 2 \ldots 4$) has little impact on $S_\mathrm{o}^\mathrm{est}$,

	Pe = 20	1 000	10 000	40 000
$K = 0$	−0.0 (2.0)	0.4 (0.4)	0.6 (13.8)	−0.7 (52.3)
$K = 2$	0.4 (2.1)	0.2 (−0.8)	0.7 (11.1)	1.1 (42.0)
$K = 4$	0.7 (0.4)	0.2 (−0.7)	0.4 (8.9)	0.8 (33.5)

TABLE I. Relative errors in the concentration ratio (eq. 8b) between the `fracture` and `matrix` regions for the cubic (random) geometries. The errors are listed as percentages. Significant error is observed for the high Péclet numbers in the random geometry. Positive values denote larger number of tracer particles in the `fracture` region than their expected equilibrium concentration.

and does not help to avoid the overestimation caused by the existence of a maximum in $S_\mathrm{o}^\mathrm{est}$ curve (Fig. 9, right).

Our findings provide additional insights into the interpretations of tracer field tests that may have a significant impact on the economics of future oil reservoir development.

The validation section of this paper neglects flow, which may affect the concentration ratios (mass balance) of non-partitioning and partitioning tracers between `fracture` and `matrix` regions. Table I provides this information. Error in the concentration ratios stays about 1% in the cubic geometry for all values of the simulation parameters. In random geometry, significant errors are observed for Pe $\geq$ 10000. This error behavior can be attributed to the high *cell Péclet number*, $\mathrm{Pe}_\mathrm{cell}$, quantifying dominance of advection over diffusion at a given discretization level. For cell Péclet number, the size of elementary mesh element (which is 1) is taken as the characteristic length. Simulations with Pe = 40000 and the discretization resolution of ~ 256 voxels per matrix block result in $\mathrm{Pe}_\mathrm{cell} = 40000/256 \approx 156$. Similarly, $\mathrm{Pe}_\mathrm{cell} = 10000/256 \approx 39$ for Pe = 10000. This value is high relative to those suggested by Maier *et al.*[66] and the references therein, in which the upper limit was considered to be $\mathrm{Pe}_\mathrm{cell} \approx 20$. We note that these authors discussed $\mathrm{Pe}_\mathrm{cell}$ in the context of hydrodynamic dispersion in packings of impermeable spheres with the porosity of about 40%. Here we have a different numerical scenario of tracer dispersion in a low-porosity fractured system with mass transfer between the `fracture` and `matrix` regions. Therefore, direct comparison of restrictions on $\mathrm{Pe}_\mathrm{cell}$ is not appropriate. However, Table I suggests that the matrix boundaries that are not aligned with the underlying lattice, as in random geometry, and high flow rates ($\mathrm{Pe}_\mathrm{cell}$ at least 40) introduce significant concentration error even in the case of specular reflection boundary condition. This concentration error results in higher close-to-asymptotic values of estimated oil saturation $S_\mathrm{o}^\mathrm{est}$ for random geometry compared with cubic one (two bottom right panels of Fig. 9). Increase of discretization resolution to reduce $\mathrm{Pe}_\mathrm{cell}$ will change the exact shape of estimated oil saturation curves for random geometry, but will not change our qualitative conclusions.

IV. CONCLUSIONS

Our practical goal was to evaluate the industry standard estimates of remaining oil saturation in the heterogeneous carbonate reservoirs. With our model, we quantified that the commonly used expression (11) may overestimate the true oil saturation with relative error of 50–70%, depending on the injector–producer distance. For the infinitely long reservoirs, industry-standard model of tracer tests holds.

Our model relies on the three-dimensional fractured geometries generated with Voronoi tessellations. A set of seed points is introduced, Voronoi tessellation on them is performed, and after shrinking each Voronoi cell the released void space is considered as "fractures." Each Voronoi cell represents rock matrix. Using two spatial arrangements of seed points, cubic (regular) and random, we have created two fractured geometries with 2304 Voronoi cell each. These geometries are meshed using a high-performance code at a spatial resolution of 256 mesh nodes per $V^{1/3}$, where V is the volume of Voronoi cell. This mesh is then used to simulate water flow with the lattice-Boltzmann method and water-tracer dispersion with the random-walk particle-tracking method.

Simulation of tracer dispersion deals with the diffusion of tracers in `fracture` and `matrix` regions, flow in the `fracture` region, and mass transfer between the `fracture` and `matrix` regions. The `matrix` region consists of rock mass, oil and water. The tracers are allowed to have different partitioning coefficients, or the ratios between the equilibrium concentrations of the tracers in oil and water. We have performed an in-depth validation of the correct mass-transfer behavior, for both time-dependent and equilibrium profiles close to the `fracture`–`matrix` interface. This validation has demonstrated acceptable accuracy of the employed specular reflection boundary condition, including such non-trivial cases as corners in three dimensions.

The described numerical approach was employed to represent the field-scale tracer tests used to estimate remaining oil saturation, S_o, in an oil reservoir. Our approach allows to i) prescribe an input oil saturation value $S_\mathrm{o}^{\mathrm{est},*}$, and ii) estimate $S_\mathrm{o}^\mathrm{est}$ from the tracer concentration histories detected in producing wells at different locations downstream. We observe significant variation of the estimated oil saturation with the distance traveled by the tracers. This distance is comparable or exceeds a typical well spacing in a Middle East oil reservoir when realistic flow rates are considered. For the considered simulation parameters (two geometries, two partitioning coefficients) with input oil saturation of $S_\mathrm{o}^\mathrm{est} = 0.161$ the model predicts i) the transient values of $S_\mathrm{o}^\mathrm{est}$ up to ≈ 0.28, and ii) the accurate value of oil saturation of $S_\mathrm{o}^\mathrm{est} \approx 0.161$ for the infinitely-long injector–producer distance. The overestimation of $S_\mathrm{o}^\mathrm{est}$ is caused by the existence of a maximum in $S_\mathrm{o}^\mathrm{est}$ along the injector–producer distance curve. This maximum starts to develop at the flow rates significantly below the field-scale ones.

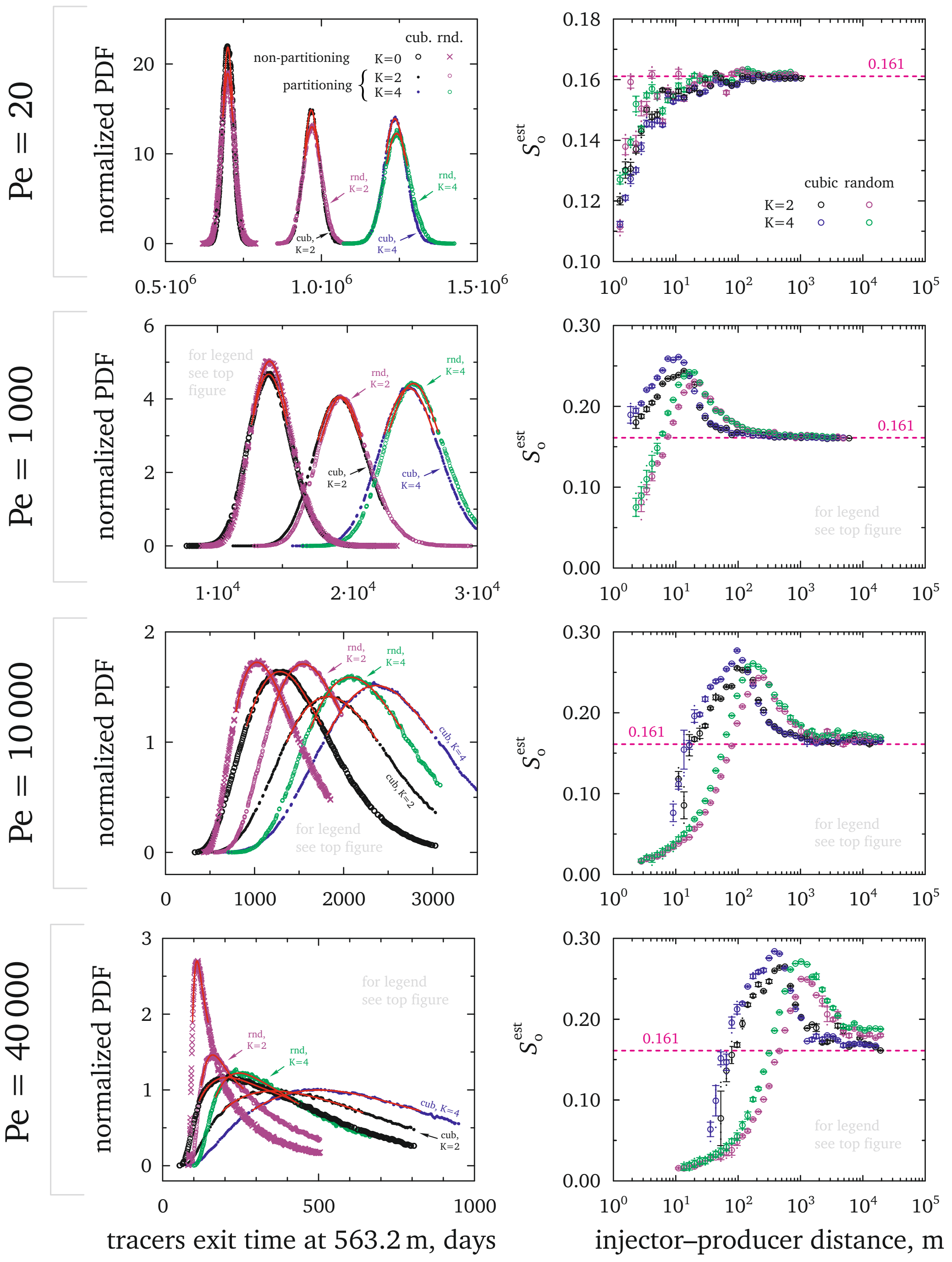

FIG. 9. The left-hand figure panels show the distribution of tracer exit times for the injector–producer distance of 563.2 m. The red lines near top of each curve are the 4-th order polynomial fits for accurate localization of each distribution maximum. The areas under each distribution are normalized to one, considering only shown points. The right-hand column shows estimated oil saturation S_o^{est} using eq. (11) at various distances between tracer injection and detection. Each value of S_o^{est} is shown together with its standard error (95% confidence interval). Magenta dashed line denotes the reference value of $S_o^{est} \approx 0.161$, see eq. (12). Data are shown for non-partitioning ($K = 0$) and partitioning tracers ($K = 2$ and $K = 4$), and for cubic and random geometries (Fig. 8).

For the reference oil saturation we use eq. (12) which includes the remaining oil saturation in the matrix S_o, matrix porosity ϕ_m, and fracture porosity Φ_f. While the value of ϕ_m is commonly determined experimentally using core samples, the porosity of fractures or super-k channels is known with much smaller precision. Omission or inaccurate estimation of Φ_f in eq. (12) results in $S_o^{est,*} \equiv S_o$ and, therefore, in an underestimation of the remaining oil saturation ($S_\mathrm{o}^\mathrm{est} < S_\mathrm{o}$), even in the limit of infinite injector–producer distance.

ACKNOWLEDGMENTS

This work was performed with the funding from KAUST to the Ali I. Naimi Petroleum Engineering Research Center. We would like to thank Prof. Hussein Hoteit for numerous discussions and valuable help. For computational time, this research used the resources of the Supercomputing Laboratory at King Abdullah University of Science & Technology (KAUST) in Thuwal, Saudi Arabia.

[1]G. Mikutis, C. A. Deuber, L. Schmid, A. Kittilä, N. Lobsiger, M. Puddu, D. O. Asgeirsson, R. N. Grass, M. O. Saar, and W. J. Stark, "Silica-Encapsulated DNA-Based Tracers for Aquifer Characterization," Environ. Sci. Technol. **52**, 12142–12152 (2018).
[2]M. Jin, M. Delshad, V. Dwarakanath, D. C. McKinney, G. A. Pope, K. Sepehrnoori, C. E. Tilburg, and R. E. Jackson, "Partitioning Tracer Test for Detection, Estimation, and Remediation Performance Assessment of Subsurface Nonaqueous Phase Liquids," Water Resour. Res. **31**, 1201–1211 (1995).
[3]V. Dwarakanath, N. Deeds, and G. A. Pope, "Analysis of Partitioning Interwell Tracer Tests," Environ. Sci. Technol. **33**, 3829–3836 (1999).
[4]P. E. Mariner, M. Jin, J. E. Studer, and G. A. Pope, "The First Vadose Zone Partitioning Interwell Tracer Test for Nonaqueous Phase Liquid and Water Residual," Environ. Sci. Technol. **33**, 2825–2828 (1999).
[5]M. Sanni, M. Al-Abbad, S. Kokal, Ø. Dugstad, S. Hartvig, and O. Huseby, "Pushing the Envelope of Residual Oil Measurement: A Field Case Study of a New Class of Inter-Well Chemical Tracers," (Society of Petroleum Engineers, Dubai, UAE, 2016) SPE-181324-MS.
[6]J. S. Tang and B. Harker, "Interwell Tracer Test To Determine Residual Oil Saturation In A Gas-Saturated Reservoir. Part I: Theory And Design," J. Can. Pet. Technol. **30**, 76–85 (1991).
[7]J. Vanderborght, M. Vanclooster, A. Timmerman, P. Seuntjens, D. Mallants, D. Kim, D. Jacques, L. Hubrechts, C. Gonzalez, J. Feyen, J. Diels, and J. Deckers, "Overview of inert tracer experiments in key Belgian soil types: Relation between transport and soil morphological and hydraulic properties," Water Resour. Res. **37**, 2873–2888 (2013).
[8]P. Villegas, V. Paredes, T. Betancur, J. D. Taupin, and L. E. Toro, "Groundwater evolution and mean water age inferred from hydrochemical and isotopic tracers in a tropical confined aquifer," Hydrol. Processes **32**, 2158–2175 (2018).
[9]T. C. Atkinson, D. I. Smith, J. J. Lavis, and R. J. Witaker, "Experiments in tracing underground waters in limestones," Journal of Hydrology **19**, 323–349 (1973).
[10]J. C. Giddings, *Dynamics of chromatography: principles and theory* (Marcel Dekker, 1965).
[11]G. M. Shook, S. L. Ansley, and A. Wylie, "Tracers and Tracer Testing: Design, Implementation, and Interpretation Methods," Tech. Rep. INEEL/EXT-03-01466 (Idaho National Engineering and Environmental Laboratory, 2004).
[12]B. Zemel, *Tracers in the Oil Field*, Developments in Petroleum Science (Book 43) (Elsevier Science, 1995).
[13]K. M. Gerke, M. V. Karsanina, and D. Mallants, "Universal Stochastic Multiscale Image Fusion: An Example Application for Shale Rock," Sci. Rep. **5**, 15880 (2015).
[14]M. V. Karsanina and K. M. Gerke, "Hierarchical optimization: Fast and robust multiscale stochastic reconstructions with rescaled correlation functions," Phys. Rev. Lett. , 265501 (2018).
[15]J. Warren and P. Root, "The Behavior of Naturally Fractured Reservoirs," SPE J. **3**, 245–255 (1963).
[16]H. Kazemi, L. S. Merrill, K. L. Porterfield, and P. R. Zeman, "Numerical Simulation of Water-Oil Flow in Naturally Fractured Reservoirs," SPE J. **16**, 317–326 (1976).
[17]P. Małoszewski and A. Zuber, "On the theory of tracer experiments in fissured rocks with a porous matrix," J. Hydrol. **79**, 333–358 (1985).
[18]F. de Smedt and P. Wierenga, "Mass transfer in porous media with immobile water," J. Hydrol. **41**, 59–67 (1979).
[19]S. Geiger, M. Dentz, and I. Neuweiler, "A Novel Multi-Rate Dual-Porosity Model for Improved Simulation of Fractured and Multiporosity Reservoirs," SPE J. **18**, 670–684 (2013).
[20]D. Cantrell and R. Hagerty, "Reservoir rock classification, Arab-D reservoir, Ghawar field, Saudi Arabia," Geoarabia **8**, 435 – 462 (2003).
[21]L. Mi, H. Jiang, Y. Wang, and B. Yan, "A Novel and Adaptive Mesh Method for Arbitrary Discrete Fracture Networks DFN Simulations," (Society of Petroleum Engineers, Manama, Kingdom of Bahrain, 2017) SPE-183980-MS.
[22]A. Fourno, T.-D. Ngo, B. Noetinger, and C. La Borderie, "FraC: A new conforming mesh method for discrete fracture networks," J. Comp. Phys. **448**, 713–732 (2019).
[23]T.-D. Ngo, A. Fourno, and B. Noetinger, "Modeling of transport processes through large-scale discrete fracture networks using conforming meshes and open-source software," J. Hydrol. **2017**, 66–79 (2017).
[24]S. Berrone, M. F. Benedetto, A. Borio, S. Pieraccini, and S. Scialó, "The Virtual Element Method for large scale Discrete Fracture Network simulations: fracture-independent mesh generation," Proc. Appl. Math. Mech. **15**, 19–22 (2015).
[25]S. Khirevich and T. Patzek, "Behavior of numerical error in pore-scale lattice Boltzmann simulations with simple bounce-back rule: Analysis and highly accurate extrapolation," Phys. Fluids **30**, 093604 (2018).
[26]A. Okabe, *Spatial tessellations: concepts and applications of Voronoi diagrams*, 2nd ed. (John Wiley & Sons, 2000).
[27]C. W. Passchier and R. A. J. Trouw, *Microtectonics*, 2nd ed. (Springer Berlin Heidelberg, 2005).
[28]C. An, B. Yan, M. Alfi, L. Mi, J. E. Killough, and Z. Heidari, "Estimating spatial distribution of natural fractures by changing NMR T2 relaxation with magnetic nanoparticles," J. Pet. Sci. Eng. **157**, 273–287 (2017).
[29]F. N. Rashid, *The Kometan Formation: Reservoir characteristics of tight carbonates in the Western Zagros Basin*, Ph.D. thesis, University of Leeds, England (2015), Chapter 3, p. 39.
[30]Beaver, Vicki, "Tundra polygons," (2011), accessed: 04-07-2019. `www.vickibeaver.com`.
[31]M. Wilson, "Wooster Geologists," (2011), accessed: 04-07-2019.
[32]C. Carvalho, P. Viegas, and M. Cachao, "Thalassinoides and its producer: Populations of Mecochirus buried within their burrow systems, Boca Do Chapim Formation (Lower Cretaceous), Portugal," Palaios **22**, 107–112 (2007).
[33]S. Raghavachary, "Fracture generation on polygonal meshes using voronoi polygons," in *ACM SIGGRAPH 2002 Conference Abstracts and Applications*, SIGGRAPH '02 (ACM, New York, NY, USA, 2002) pp. 187–187.
[34]E. Ghazvinian, M. Diederichs, and R. Quey, "3D random Voronoi grain-based models for simulation of brittle rock damage and fabric-guided micro-fracturing," J. Rock Mech. Eng. Geotech. Eng. **6**, 506–521 (2014).
[35]M. Adda-Bedia, R. Arias, M. Ben Amar, and F. Lund, "Dynamic Instability of Brittle Fracture," Phys. Rev. Lett. **82**, 2314–2317 (1999).
[36]S. Kumar, S. K. Kurtz, J. R. Banavar, and M. G. Sharma, "Properties of a three-dimensional Poisson-Voronoi tesselation: A Monte Carlo study," J. Stat. Phys. **67**, 523–551 (1992).
[37]V. Lucarini, "Three-Dimensional Random Voronoi Tessellations: From Cubic Crystal Lattices to Poisson Point Processes," J. Stat. Phys. **134**, 185–206 (2009).
[38]C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The QUICKHULL algorithm for convex hulls," ACM Trans. Math. Software **22**, 469–

483 (1996).

[39]A. Fourno, C. Grenier, A. Benabderrahmane, and F. Delay, "A continuum voxel approach to model flow in 3D fault networks: A new way to obtain up-scaled hydraulic conductivity tensors of grid cells," J. Hydrol. **493**, 68–80 (2013).

[40]J. Hyman, C. Gable, S. Painter, and N. Makedonska, "Conforming delaunay triangulation of stochastically generated three dimensional discrete fracture networks: A feature rejection algorithm for meshing strategy," SIAM J. Sci. Comput. **36**, A1871–A1894 (2014).

[41]C. K. Aidun and J. R. Clausen, "Lattice-Boltzmann Method for Complex Flows," Annu. Rev. Fluid Mech. **42**, 439–472 (2010).

[42]D. Maggiolo, F. Picano, and M. Guarnieri, "Flow and dispersion in anisotropic porous media: A lattice-Boltzmann study," Phys. Fluids **28**, 102001 (2016).

[43]P. L. Bhatnagar, E. P. Gross, and M. Krook, "A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems," Phys. Rev. **22**, 511–525 (1954).

[44]Y. Qian, D. d'Humières, and P. Lallemand, "Lattice BGK models for Navier-Stokes equation," Europhys. Lett. **17**, 479–484 (1992).

[45]I. Ginzburg and D. d'Humières, "Multireflection boundary conditions for lattice Boltzmann models," Phys. Rev. **68**, 066614 (2009).

[46]d'Humières, "Generalized lattice Boltzmann equations Rarefied Gas Dynamics: Theory and Simulations," Prog. Astronaut. Aeronaut. **159**, 450–458 (1992).

[47]D. d'Humières, I. Ginzburg, M. Krafczyk, P. Lallemand, and L.-S. Luo, "Multiple-relaxation-time lattice Boltzmann models in three dimensions," Phil. Trans. R. Soc. Lond. A **360**, 437–451 (2002).

[48]I. Ginzburg, F. Verhaeghe, and D. d'Humières, "Two-relaxation-time lattice Boltzmann scheme: About parametrization, velocity, pressure and mixed boundary conditions," Commun. Comput. Phys. **3**, 427–478 (2008).

[49]S. Khirevich, I. Ginzburg, and U. Tallarek, "Coarse- and fine-grid numerical behavior of MRT/TRT Lattice-Boltzmann schemes in regular and random sphere packings," J. Comp. Phys. **281**, 708–742 (2015).

[50]D. d'Humières and I. Ginzburg, "Viscosity independent numerical errors for lattice Boltzmann models: From recurrence equations to "magic" collision numbers," Comp. Math. Appl. **58**, 823840 (2009).

[51]P. Salamon, D. Fernàndez-Garcia, and Gómez-Hernández, "A review and numerical assessment of the random walk particle tracking method," J. Contam. Hydrol. **87**, 277–305 (2006).

[52]F. Delay, P. Ackerer, and C. Danquigny, "Simulating Solute Transport in Porous or Fractured Formations Using Random Walk Particle Tracking," Vadose Zone J. **4**, 360–379 (2005).

[53]J. A. Rudnick and G. D. Gaspari, *Elements of the random walk: an introduction for advanced students and researchers* (Cambridge University Press, 2004).

[54]I. M. Mazzitelli, F. Toschi, and A. S. Lanotte, "An accurate and efficient Lagrangian sub-grid model," Phys. Fluids **26**, 095101 (2014).

[55]M. Bechtold, J. Vanderborght, O. Ippisch, and H. Vereecken, "Efficient random walk particle tracking algorithm for advective-dispersive transport in media with discontinuous dispersion coefficients and water contents," Water Resour. Res. **47**, W10526 (2011).

[56]A. Daneyko, D. Hlushkou, V. Baranau, S. Khirevich, A. Seidel-Morgenstern, and U. Tallarek, "Computational investigation of longitudinal diffusion, eddy dispersion, and trans-particle mass transfer in bulk, random packings of core–shell particles with varied shell thickness and shell diffusion coefficient," J. Chrom. A **1407**, 139–156 (2015).

[57]P. Szymczak and A. J. C. Ladd, "Boundary conditions for stochastic solutions of the convection–diffusion equation," Phys. Rev. E **68**, 036704 (2003).

[58]S. Khirevich, *High-performance computing of flow, diffusion, and hydrodynamic dispersion in random sphere packings*, Ph.D. thesis, Philipps-Universität Marburg, Germany (2010).

[59]E. J. Scott, L. H. Tung, and H. G. Drickamer, "Diffusion through an Interface," J. Chem. Phys. **19**, 1075–1078 (1951).

[60]T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun Guide* (Pafnuty Publications, 2014).

[61]C. Agca, G. Pope, and K. Sepehrnoori, "Modelling and analysis of tracer flow in oil reservoirs," J. Pet. Sci. Eng. **4**, 3–19 (1990).

[62]C. E. Cooke, "Method of determining fluid saturations in reservoirs," (1971), uS Patent No. 3590923.

[63]K. J. Cunningham, M. Sukop, H. Huang, P. F. Alvarez, H. Allen Curran, R. A. Renken, and J. F. Dixon, "Prominence of ichnologically-influenced macroporosity in the karst Biscayne aquifer: Stratiform "super-K" zones," Geol. Soc. Am. Bull. **121**, 164–180 (2009).

[64]S. C. Schon and J. W. Head, "Super-permeability zones and the formation of outflow channels on Mars," (Lunar and Planetary Institute, League City, Texas, 2007).

[65]S. O. Viig, H. Juilla, P. Renouf, R. Kleven, B. Krognes, O. Dugstad, and O. Huseby, "Application of a New Class of Chemical Tracers To Measure Oil Saturation in Partitioning Interwell Tracer Tests," (Society of Petroleum Engineers, Woodlands, Texas, USA, 2013) SPE-164059-MS.

[66]R. S. Maier, D. M. Kroll, R. S. Bernard, S. E. Howington, J. F. Peters, and H. T. Davis, "Pore-scale simulation of dispersion," Phys. Fluids **12**, 2065–2079 (2000).