13
Flow Diagnostics

It is rarely sufficient to study well responses and pressure and saturation distributions to understand the communication patterns in a complex reservoir model. To gain a better qualitative picture, you typically want understand from what region a given producer drains. To what region does a given injector provide pressure support? Which injection and production wells are in communication? Which parts of the reservoir affect this communication? How much does each injector support the recovery from a given producer? Do any of the wells have backflow? What is the sweep and displacement efficiency within a given drainage, sweep, or well-pair region? Which regions are likely to remain unswept? Likewise, you may also want to perform what-if and sensitivity analyses to understand how different parameters and their inherent uncertainty affect reservoir responses.

Simulating a full reservoir model containing a comprehensive description of geology, reservoir fluids, flow physics, well controls, and coupling to surface facilities is a computationally demanding task that may take hours or even days to complete. This limits your ability to study parameter variations and is at odds with the philosophy of modern reservoir characterization techniques, which often generate hundreds of equiprobable realizations to quantify uncertainty in the characterization. In this chapter, we therefore introduce a set of simple techniques, referred to as *flow diagnostics*, which you can use to delineate volumetric communications and improve your understanding of how flow patterns in the reservoir are affected by geological heterogeneity and respond to engineering controls. In their basic setup, these techniques only assume that you have a geocellular model of the reservoir and rely on the solutions of single-phase, incompressible flow problems, as discussed in Chapters 4 and 5. However, you can equally well employ the techniques to analyze representative flow field extracted from multiphase simulations of the type discussed in Chapters 10–12.

My definition of flow diagnostics is *a family of simple and controlled numerical flow experiments that are run to probe a reservoir model, establish connections and basic volume estimates, and quickly provide a qualitative picture of the flow patterns in the reservoir and quantitative measures of the heterogeneity in dynamic flow paths.* You can also use these methods to compute quantitative information about the recovery process in settings somewhat simpler than what would be encountered in an actual field. Using flow diagnostics, you can rapidly and iteratively perturb simulation input and evaluate the

https://doi.org/10.1017/9781108591416.018 Published online by Cambridge University Press

resulting changes in volumetric connections and communications to build an understanding of cause and effects in your model. The following presentation is inspired by [274] and [218], who developed the concept of flow diagnostics based on standard finite-volume discretizations for time-of-flight and tracer partitions [221, 98]. Similar ideas have previously been used within streamline simulation [79] for ranking and upscaling [140, 27, 278], identifying reservoir compartmentalization [127], rate optimization [287, 244, 144], and flood surveillance [32].

Flow diagnostics is inexpensive to compute, and I generally recommend that you use these methods to prescreen your models before you start to conduct more comprehensive multiphase simulations. MRST already offers a few simple interactive GUIs to this end, and I expect that similar capabilities will appear in commercial tools in the not too distant future. Interactive tools are not easy to present in book form, and if you want to see these work in practice, you should try out some of the examples that follow MRST, or study the exercises presented throughout this chapter. Much of the same ideas can also be used to postprocess multiphase flow simulations; such capabilities are currently available in the OPM ResInsight viewer (http://resinsight.org). Early research [218, 171] indicates that flow diagnostics offers a computationally inexpensive complement (or alternative) to full-featured multiphase simulations to provide flow information in various reservoir management workflows, e.g., as inexpensive objection functions for well placement and rate optimization, or as a means to perform what-if and sensitivity analyzes in parameter regions surrounding preexisting simulations.

13.1 Flow Patterns and Volumetric Connections

You have already been introduced to the basic quantities that lie at the core of flow diagnostics in Chapters 4 and 5: as you probably recall from Section 4.3.3, we can derive time lines that show how heterogeneity affects flow patterns for an instantaneous velocity field $\vec{v}$ by computing

- the *forward time-of-flight*, defined by

$$\vec{v} \cdot \nabla \tau_f = \phi, \qquad \tau_f|_{\text{inflow}} = 0, \tag{13.1}$$

  which measures time it takes a neutral particle to travel to a given point in the reservoir from the nearest fluid source or inflow boundary; and
- the *backward time-of-flight*, defined by

$$-\vec{v} \cdot \nabla \tau_b = \phi, \qquad \tau_b|_{\text{outflow}} = 0, \tag{13.2}$$

  which measures the time it takes a neutral particle to travel from a given point in the reservoir to the nearest fluid sink or outflow boundary.

The sum of the forward and backward time-of-flight at a given point in the reservoir gives the total *residence time* of an imaginary particle as it travels from the nearest fluid source or inflow boundary to the nearest fluid sink or outflow boundary.

Studying iso-contours of time-of-flight gives you a time map for how more complex multiphase displacements may evolve under fixed well and boundary conditions and reveals more information about the flow field than pressure and velocities alone. We illustrated this already in Chapter 5, where Figure 5.3 on page 160 showed time lines for a quarter five-spot flow pattern, and the total residence time was used to distinguish high-flow regions from stagnant regions. Likewise, in Figure 5.11 on page 173 we used time-of-flight to identify non-targeted regions for a complex field model; that is, regions with high τ_f values that were likely to remain unswept and hence would be obvious targets to investigate for placement of additional wells. From time-of-flight, you can derive various measures of dynamic heterogeneity, as we will see in the next section, or compute proxies of economical measures such as net-present value for multiphase flow models; see [218, 171].

In a similar manner, we can determine all points in the reservoir that are influenced by a given fluid source or inflow boundary by solving the following equation to compute the *volumetric injector partition*:

$$\vec{v} \cdot \nabla c_i = 0, \qquad c_i|_{\text{inflow}} = 1. \tag{13.3}$$

To understand what this equation represents, let us think of an imaginary painting experiment in which we inject a tracer dye (a massless, non-diffusive ink) of a unique color at each fluid source or point on the inflow boundary we want to trace the influence from. The ink will start flowing through the reservoir and paint every point it contacts. Eventually, the fraction of different inks that flows past each point in the reservoir reaches a steady state, and by measuring these fractions, we can determine the extent to which each different ink influences a specific point and find the *influence region* of an inflow point. Likewise, to determine how much the flow through each point is influenced by a fluid sink or point on the outflow boundary, we reverse the flow field and solve similar equations for producer (or outflow) partitions,

$$-\vec{v} \cdot \nabla c_p = 0, \qquad c_p|_{\text{outflow}} = 1. \tag{13.4}$$

To summarize, flow diagnostics requires three computations: (i) solution of a pressure equation, or extraction of fluxes from a previous computation, to determine the bulk fluid movement; (ii) solution of a set of steady advection equations (13.3)–(13.4) to partition the model into influence or volumetric flow regions; and (iii) solution of time-of-flight equations (13.1)–(13.2) to give averaged time lines that describe the flow within each region. This chapter describes how you can combine and process tracer partitions and time-of-flight to gain more insight into flow patterns and volumetric connections in the reservoir. In more advanced methods, we also use *residence-time distributions* over all flow paths and analogies with tracer analysis [277, 139] for more accurate analysis of flow paths.

13.1.1 Volumetric Partitions

If each source of inflow (outflow) is assigned a unique "color" value, the resulting color concentration should in principle produce a *partition of unity* in all parts of the reservoir that are in communication with the sources of inflow (outflow). In practice, you may not be able to obtain an exact partition of unity because of numerical errors. Based on the inflow/outflow partitions, we can further define:

- *drainage regions* – each such region represents the reservoir volume that eventually will be drained by a given producer (or outflow boundary), provided that the current flow field $\vec{v}$ prevails until infinity;
- *sweep regions* – each such region represents the reservoir volume that eventually will be swept by a given injector (or inflow boundary) if the current flow conditions remain forever;
- *well pairs* – injectors and producers in communication with each other;
- *well-pair regions* – regions of the reservoir in which the flow between a given injector and producer takes place.

When visualizing multiple drainage or sweep regions in 3D, it is common to assign a unique region to each cell determined by a majority vote over all “colors” that affect the cell. Well pairs are determined by finding all injectors whose concentration is positive in one of the well completions of a given producer (or vice versa). Well-pair regions are found by intersecting injector and producer partitions, possibly followed by a majority vote. Well allocation factors will be discussed in more detail in Section 13.1.3 and in Chapter 15, in conjunction with single-phase upscaling.

Our default choice would be to assign a unique “color” to each injector and producer, but you can also subdivide wells into multiple segments and trace the influence of each segment separately. You can, for instance, use this to determine if a (horizontal) well has cross-flow, so that fluid injected in one part of the well is drawn back into the wellbore in another part of the well, or fluids produced in one completion are pushed out again in another.

We have already encountered `computeTimeOfFlight` for computing time-of-flight in Section 5.3. The `diagnostics` module additionally offers

```
D  = computeTOFandTracer(state, G, rock, 'wells', W, ...)
```

that computes forward and backward time-of-flight, injector and producer coloring, as well as sweep and drainage regions in one go for models with flow driven by wells. These quantities are represented as fields in the structure `D`:

- `inj` and `prod` give the indices for the injection and production wells in the well structure `W`;
- `tof` is a $2 \times n$ vector with τ_f in its first and τ_b in its second column;
- `itracer`/`ptracer` contain color concentrations for the injectors and producers;
- `ipart` and `ppart` hold the partitions resulting from a majority vote.

Obviously, you can associate similar quantities with boundary conditions and/or source terms. (MRST does not yet fully implement this.) Well pairs are identified by analyzing the

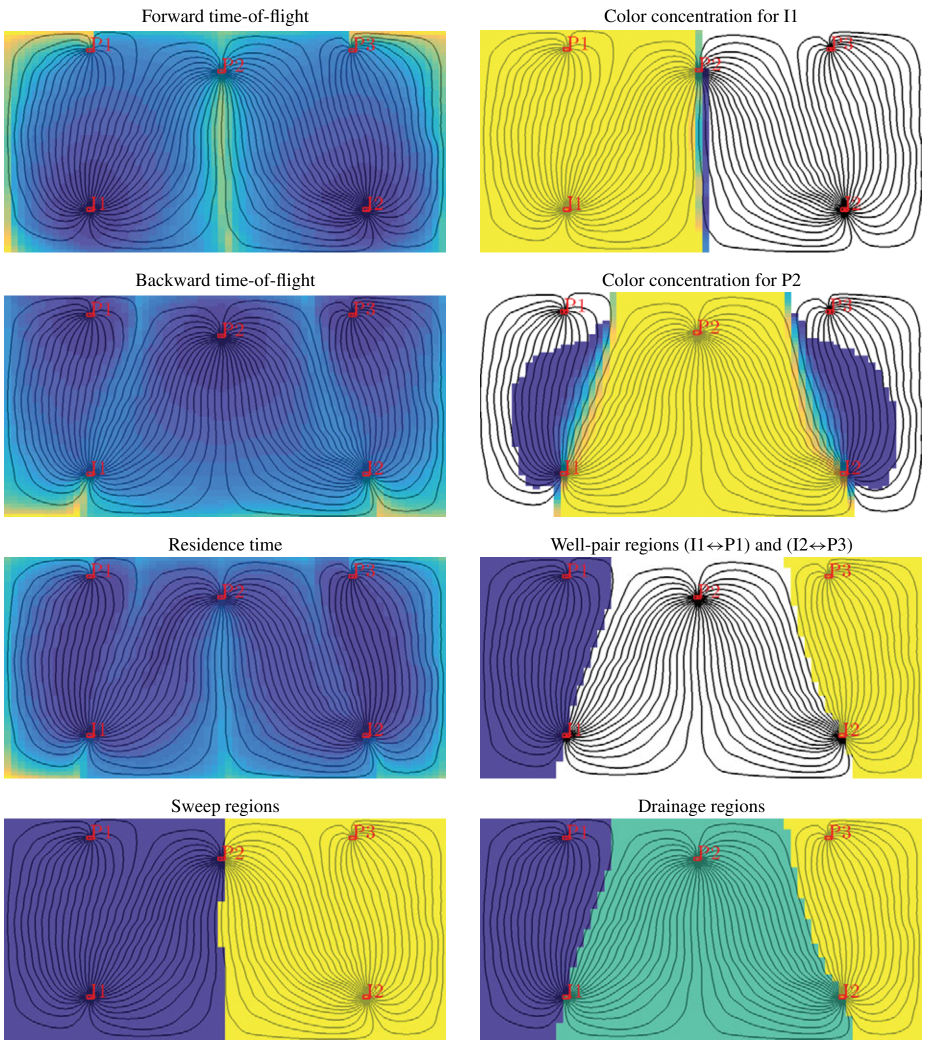

Figure 13.1 Time-of-flight, color concentration, and various types of volumetric delineations for a simple case with two injectors and three producers. Source code necessary to generate the plots is given in the `showDiagnostBasic` tutorial in the `book` module.

`itracer` and `ptracer` fields. The following function also computes the pore volume of the region of the reservoir associated with each pair:

```
WP = computeWellPairs(G, rock, W, D)
```

Figure 13.1 shows time-of-flight, color concentrations, volumetric partitions, and a few combinations thereof for a simple flow problem with two injectors and three production wells. There are many other ways you can combine and plot these basic quantities to reveal volumetric connection and provide enhanced insight into flow patterns. You can, for instance, combine sweep regions with time-of-flight to provide a simple forecast of *contacted volumes* or to visualize how displacement fronts move through the reservoir. To this end, you would typically also include estimates of characteristic wave-speeds from multiphase displacement theory as discussed in Section 8.4.

13.1.2 Time-of-Flight Per Partition Region: Improved Accuracy

It is important to understand that the time-of-flight values computed by a finite-volume method like the one discussed in Section 4.4.3 are *volume-average values*, which cannot be compared directly with the *point values* you obtain by tracing streamlines, as described in Section 4.3.3. Because time-of-flight is a quantity defined from a global line integral (see (4.39)), the point-wise variations can be large inside a single grid cell, in particular near flow divides and in the near-well regions, where both high-flow and low-flow streamlines converge. To improve the accuracy of the time-of-flight within each well-pair region, we can use the color concentrations to recompute the time-of-flight values for influence region number k,

$$\vec{v} \cdot \nabla\left(c_i^k \tau_f^k\right) = c_i^k \phi. \tag{13.5}$$

In MRST, this is done by passing the option `computeWellTOFs` to the time-of-flight solver, using a call that looks something like

```
T = computeTimeOfFlight(state, G, rock, 'wells', W, ...
                        'tracer',{W(inj).cells},'computeWellTOFs', true);
```

which appends one extra column for each color at the end of return parameter `T`. You can also use a higher-order discretization, as discussed in [221, 261].

13.1.3 Well Allocation Factors

Apart from a volumetric partition of the reservoir, one is often interested in knowing the fraction of the inflow to a given producer that can be attributed to each of the injectors, or conversely, how the push from a given injector is distributed to the different producers. We refer to this as *well allocation factors*. These factors can be further refined so that they also describe the cumulative flow from the toe to the heel of the well. By computing the cumulative flux along the wellbore and plotting this flux as a function of the distance from the toe (with the flux on the x-axis and distance on the y-axis) we get a plot that is reminiscent of the plot from a *production logging tool*.

To formally define well allocation factors, we use the notation from Section 4.4.2, so that $\boldsymbol{x}[c]$ denotes the value of vector $\boldsymbol{x}$ in cell c. Next, we let $\boldsymbol{c}_n^i$ denote the injector color concentration associated with well (or segment) number n, let $\boldsymbol{c}_m^p$ denote the producer concentration associated with well number m, $\boldsymbol{q}$ the vector of well fluxes, and $\{w_k^n\}_k$ the cells in which well number n is completed. Then, the cumulative factors are defined as

$$
\begin{aligned}
a^i_{nm}[w^n_\ell] &= \sum\nolimits_{k=1}^{\ell} \boldsymbol{q}[w^n_k]\,\boldsymbol{c}^p_m[w^n_k],\\
a^p_{mn}[w^m_\ell] &= \sum\nolimits_{k=1}^{\ell} \boldsymbol{q}[w^m_k]\,\boldsymbol{c}^i_n[w^m_k].
\end{aligned} \tag{13.6}
$$

The total well allocation factor equals the cumulative factor evaluated at the heel of the well. Well allocation factors computed by `computeWellPair` are found as two arrays of structs, `WP.inj` and `WP.prod`, which give the allocation factors for all the injection and production wells (or segments) accounted for in the flow diagnostics. In each struct, the array `alloc` gives the a_{nm} factors, whereas influx or outflux that cannot be attributed to another well or segment is represented in the array `ralloc`.

13.2 Measures of Dynamic Heterogeneity

Whereas primary recovery can be reasonably approximated using averaged petrophysical properties, secondary and tertiary recovery are strongly governed by the intrinsic variability in rock properties and geological characteristics. This variability, which essentially can be observed at all scales in the porous medium, is commonly referred to as *heterogeneity*. As we have seen in previous chapters, both the rock's ability to store and transmit fluids are heterogeneous. However, it is the heterogeneity in permeability that has the most pronounced effect on flow patterns and volumetric connections in the reservoir. The importance of heterogeneity has been recognized from the earliest days of petroleum production, and over the years a number of static measures have been proposed to characterize heterogeneity, such as flow and storage capacity, Lorenz coefficient, Koval factor, and Dykstra–Parson's permeability variation coefficient, to name a few; see, e.g., [176] for a more comprehensive overview.

In the following, we discuss how some of the static heterogeneity measures from classical sweep theory can be reinterpreted in a dynamic setting if we calculate them from the time-of-flight (and color concentrations) associated with an instantaneous flow field [278]. Static measures describe the spatial distribution of permeability and porosity, and large static heterogeneity means that there are large (local) variations in the rock's ability to store and transmit fluids. *Dynamic heterogeneity measures*, on the other hand, describe the distribution of flow-path lengths and connection structure, and large heterogeneity values show that there are large variations in travel and residence times, which again tends to manifest itself in early breakthrough of injected fluids. Experience has shown that these measures, and particularly the dynamic Lorenz coefficient, correlate very well with forecasts of hydrocarbon recovery predicted by more comprehensive flow simulations. They can thus be used as effective flow proxies in various reservoir management workflows; see [278, 274, 218].

13.2.1 Flow and Storage Capacity

Dynamic heterogeneity measures are computed from the total residence time, which in turn is computed as the sum of forward and backward time-of-flight. To define dynamic flow and

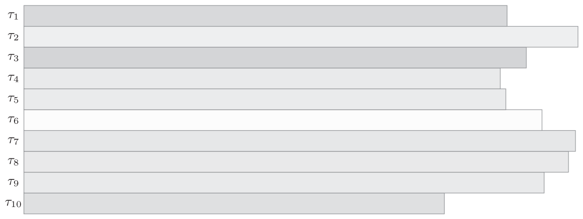

Figure 13.2 Streamtube analogue used to define dynamic flow and storage capacity. Colors illustrate different average porosities.

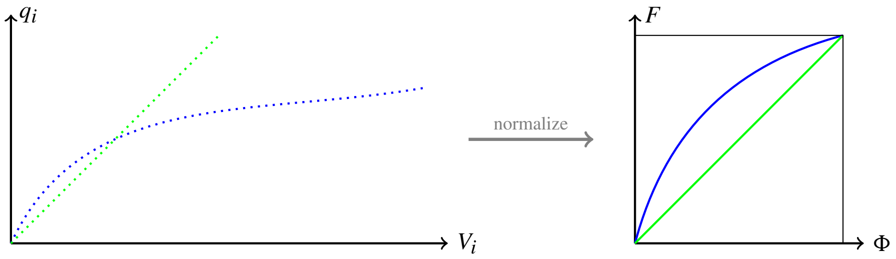

Figure 13.3 Construction of the F–Φ diagram. The plot to the left shows flow rates q_i plotted as function of streamtube volumes V_i for a homogeneous displacement (green) and for a heterogeneous displacement (blue). The right plot shows the corresponding F–Φ diagrams, where the flow rates and the streamtube volumes have been normalized.

storage capacity, we can think of the reservoir as a set of N non-communicating volumetric flow paths (streamtubes) that each has a volume V_i, a flow rate q_i, and a residence time $\tau_i = V_i/q_i$. The streamtubes are sorted so that their residence times are ascending, $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_N$; see Figure 13.2. Inside each streamtube, we assume a piston type displacement; think of a blue fluid pushing a red fluid from the left to the right in the figure. We then define the normalized flow capacity F_i and storage capacity Φ_i by,

$$\Phi_i = \sum_{j=1}^{i} V_j \Big/ \sum_{j=1}^{N} V_j, \qquad F_i = \sum_{j=1}^{i} q_j \Big/ \sum_{j=1}^{N} q_j. \tag{13.7}$$

Here, Φ_i is the volume fraction of all streamtubes that have broken through at time τ_i and F_i represent the corresponding fractional flow, i.e., the fraction of the injected fluid to the total fluid being produced. These two quantities can be plotted in a diagram as shown in Figure 13.3. From this diagram, we can also define the fractional recovery curve defined as the ratio of inplace fluid produced to the total fluid being produced; that is $(1 - F)$ plotted versus dimensionless time $t_D = d\Phi/dF$ measured in units of pore volumes injected.

To understand how the F–Φ diagram can be seen as a measure of dynamic heterogeneity, we first consider the case of a completely homogeneous displacement, for which all streamtubes will break through at the same time τ. This means that, $(\Phi_i - \Phi_{i-1})/(F_i - F_{i-1}) \propto V_i/q_i$ is constant, which implies that $F = \Phi$, since both F and Φ are normalized quantities. Next, we consider a heterogeneous displacement in which all the streamtubes have the same flow rate q. Since the residence times τ_i form a monotonically increasing sequence, we have that $\{V_i\}$ will also be monotonically increasing. In general, $F(\Phi)$ is a concave function, where the steep initial slope corresponds to high-flow regions giving early breakthrough, whereas the flat trailing tail corresponds to low-flow and stagnant regions.

In the continuous case, we can (with a slight abuse of notation) define the storage capacity as

$$\Phi(\tau) = \int_0^\tau \phi(\vec{x}(s))\,ds, \tag{13.8}$$

where $\vec{x}(\tau)$ represents all streamlines whose total travel time equals τ. By assuming incompressible flow, we have that pore volume equals the flow rate times the residence time, $\phi = q\tau$, and hence we can define the flow capacity as

$$F(\tau) = \int_0^\tau q(\vec{x}(s))\,ds = \int_0^\tau \frac{\phi(\vec{x}(s))}{s}\,ds. \tag{13.9}$$

From this, we can define *normalized, dynamic* flow and storage capacities by

$$\hat{\Phi}(\tau) = \frac{\Phi(\tau)}{\Phi(\infty)}, \qquad \hat{F}(\tau) = \frac{F(\tau)}{F(\infty)}.$$

Henceforth, we only discuss the normalized quantities and for simplicity we drop the hat symbol. To compute these quantities for a discrete grid model, we would have to first compute a representative set of streamlines, associate a flow rate, a pore volume, and a total travel time to each streamline, and then compute the cumulative sums as in (13.7).

Next, we consider how these concepts carry over to our flow diagnostics setting with time-of-flight computed by a finite-volume method and not by tracing streamlines. Let `pv` be an $n \times 1$ array containing pore volumes of the n cells in the grid and `tof` be an $n \times 2$ array containing the forward and backward time-of-flights. We can now compute the cumulative, normalized storage capacity `Phi` as follows:

```
t        = sum(tof,2);      % total travel time
[ts,ind] = sort(t);         % sort cells based on travel time
v        = pv(ind);         % put pore volumes in correct order
Phi      = cumsum(v);       % cumulative sum
vt       = full(Phi(end));  % total volume of region
Phi      = [0; Phi/vt];     % normalize to units of pore volumes
```

We can compute the flow rate for each cell directly as the ratio between pore volume and residence time if we assume incompressible flow. With this, the `computeFandPhi` calculates the cumulative, normalized flow capacity `F` as follows

```
q      = v./ts;           % back out flux based on incompressible flow
ff     = cumsum(q);       % cumulative sum
ft     = full(ff(end));   % total flux computed
F      = [0; ff/ft];      % normalize and store flux
```

The result of the above calculation is that we have two sequences Φ_i and F_i that are both given in terms of the residence time τ_i. If we sort the points (Φ_i, F_i) according to ascending values of τ_i, we obtain a sequence of discrete points that describe a parametrized curve in 2D space. The first end point of this curve is at the origin: If no fluids have entered the domain, the cumulative flow capacity is obviously zero. Likewise, full flow capacity is reached when the domain is completely filled, and since we normalize both Φ and F by their value at the maximum value of τ, this corresponds to the point (1,1). Given that both F and Φ increase with increasing values of τ, we can use linear interpolation to define a continuous, monotonic, increasing function $F(\Phi)$.

13.2.2 Lorenz Coefficient and Sweep Efficiency

The *Lorenz coefficient* is a popular measure of heterogeneity and is defined as the difference in flow capacity from that of an ideal piston-like displacement:

$$L_c = 2\int_0^1 \big(F(\Phi) - \Phi\big) d\Phi. \tag{13.10}$$

In other words, the Lorenz coefficient is equal twice the area under the $F(\Phi)$ curve and above the line $F = \Phi$, and has values between zero for homogeneous displacement and unity for an infinitely heterogeneous displacement; see Figure 13.4. Assuming that the

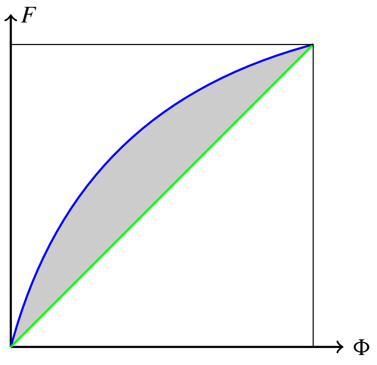

Figure 13.4 The definition of Lorenz coefficient from the F–Φ diagram. The green line represents a completely homogeneous displacement in which all flow paths have equal residence times. Te blue line is a heterogeneous displacement with significant variation in residence times. The Lorenz coefficient is defined as two times the gray area.

flow and storage capacity are given as two vectors F and Phi, the Lorenz coefficient can be computed by applying a simple trapezoid rule

```
v  = diff(Phi,1);
Lc = 2*(sum((F(1:end-1)+F(2:end))/2.*v) - .5);
```

This is implemented in the computeLorenz function in the diagnostics module.

We can also use the F–Φ diagram to compute the *volumetric sweep efficiency* E_v, which measures how efficient injected fluids are used. Here, E_v denotes the volume fraction of inplace fluid that has been displaced by injected fluid, or equivalently, the ratio between the volume contacted by the displacing fluid at time t and the volume contacted at time $t = \infty$. In our streamtube analogue, only streamtubes that have not yet broken through will contribute to sweep the reservoir. Normalizing by total volume, we thus have

$$\begin{aligned}
E_v(t) &= \frac{q}{V}\int_0^t \big[1 - F\big(\Phi(\tau)\big)\big]\,d\tau \\
&= \frac{qt}{V} - \frac{q}{V}\int_0^t F(\tau)\,d\tau = \frac{qt}{V} - \frac{q}{V}\Big[F(t)t - \int_0^F \tau\,dF\Big] \\
&= \frac{t}{\bar\tau}\big(1 - F(t)\big) + \frac{1}{\bar\tau}\int_0^\Phi \bar\tau\,d\Phi = \Phi + (1-F)\frac{d\Phi}{dF} = \Phi + (1-F)t_D.
\end{aligned}$$

The third equality follows from integration by parts, and the fourth equality since $\bar\tau = V/q$ and $\tau\,dF = \bar\tau\,d\Phi$. Here, the quantity $d\Phi/dF$ takes the role as dimensionless time. Prior to breakthrough, $E_v = t_D$. After breakthrough, Φ is the volume of fully swept flow paths, whereas $(1-F)t_D$ is the volume of flow paths being swept.

The implementation in MRST is quite simple and can be found in the utility function computeSweep. Starting from the two arrays F and Phi, we first remove any flat segments in F to avoid division by zero

```
inz        = true(size(F));
inz(2:end) = F(1:end-1)~=F(2:end);
F          = F(inz);
Phi        = Phi(inz);
```

Then, dimensionless time and sweep efficiency can be computed as follows:

```
tD = [0; diff(Phi)./diff(F)];
Ev = Phi + (1-F).*tD;
```

Figure 13.5 shows the F–Φ and the sweep diagram for the simple example with two injectors and three producers from Figure 13.1. For this particular setup, the Lorenz coefficient is approximately 0.25, which indicates that we can expect a mildly heterogeneous displacement with some flow paths that break through early and relatively small stagnant regions. From the diagram of the sweep efficiency, we see that 70% of the fluids in place can be produced by injecting one pore volume. By injecting two additional pore volumes almost all the in-place fluid can be produced.

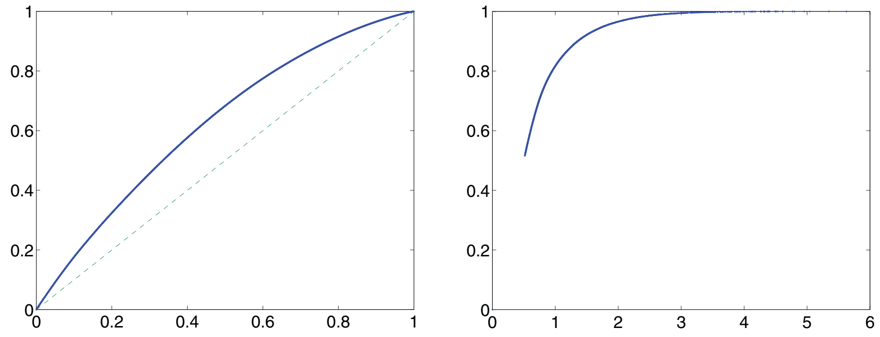

Figure 13.5 F–Φ and sweep diagram for the simple case with two injectors and three producers, which has a Lorenz coefficient of 0.2475.

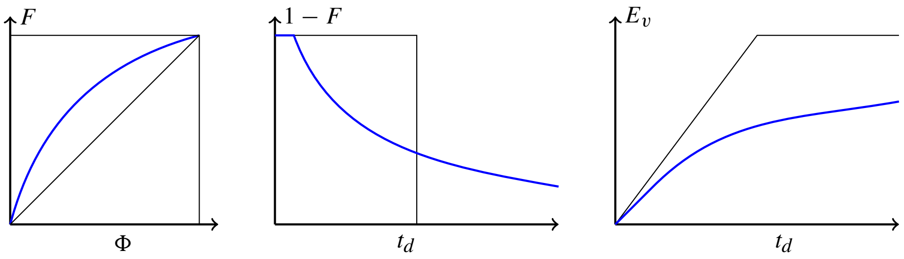

Figure 13.6 The three basic flow diagnostics curves: F–Φ diagram, fractional recovery curve, and sweep efficiency. All quantities Φ, F, E_v, and t_D are dimensionless; t_D is given in terms of pore volumes injected (PVI).

Altogether, we have defined three different curves that can be derived from the residence/travel time. The curves shown in Figure 13.6 are visually intuitive and emphasize different characteristics of the displacement:

- The F–Φ curve is useful for assessing the overall level of displacement heterogeneity. The closer this curve is to a straight line, the better is the displacement.
- The fractional recovery curve emphasizes early-time breakthrough behavior and can have utility as a proxy for fractional recovery of the fluid in place.
- Sweep efficiency highlights the behavior after breakthrough and has utility as a proxy for recovery factor.

The curves can be defined for the field as a whole, or be associated with sector models, individual swept volumes, well-pair regions, and so on.

Whereas visually intuitive information is useful in many workflows, others need measures defined in terms of real numbers. In our work, we have primarily used the Lorenz coefficient, but dynamic analogues of other classical heterogeneity measures can be defined in a similar fashion. For instance, the dynamic Dykstra–Parsons' coefficient is defined as

$$V_{DP} = \frac{(F')_{\Phi=0.5} - (F')_{\Phi=0.841}}{(F')_{\Phi=0.5}}, \tag{13.11}$$

where $\Phi = 0.5$ corresponds to the mean storage capacity, while $\Phi = 0.841$ is the mean value plus one standard deviation. Likewise, we can define the dynamic flow heterogeneity index,

$$F_{HI} = F(\Phi^*)/\Phi^*, \qquad F'(\Phi^*) = 1. \tag{13.12}$$

One can show that $(\frac{dF}{d\Phi})_i = \frac{\bar{\tau}}{\tau_i}$, where $\bar{\tau} = V/q = \sum V_i / \sum q_i$ is the average residence time for all the streamtubes. These heterogeneity measures have not yet been implemented in the `diagnostics` module.

Computer exercises

13.2.1 Implement Dykstra–Parsons' coefficient (13.11) and the flow heterogeneity index (13.12) as new functions for the diagnostics module.

13.2.2 Use time-of-flight values defined per influence region as discussed in Section 13.1.2 on page 482 to implement refined versions of the dynamic heterogeneity measures.

13.2.3 Compute heterogeneity measures for each well pair in the model with two injectors and three producers. (Original source code: `showDiagnostBasics`.) Are there differences between the different regions?

13.2.4 Try to make the displacement shown in Figures 13.1 and 13.5 less heterogeneous by moving the wells and/or by changing the relative magnitude of the injection/production rates.

13.2.5 Define a set of multiphase flow simulations cases, e.g., one five-spot simulation for each layer of the SPE 10 model (see Section 13.4.2). Which of the heterogeneity measures above has the strongest correlation with the total hydrocarbon recovery from multiphase simulations?

13.3 Residence-Time Distributions

You have previously seen how we can compute point-wise values of time-of-flight (and influence regions) by tracing streamlines or compute cell-averaged values by solving a steady, linear transport problem using a finite-volume method. This section sheds more light into the finite-volume approach and discusses how we can overcome some of its potential limitations. The discussion follows [171] closely and presents functionality that is new in MRST 2018b.

We assume incompressible flow to simplify the discussion and start by writing the finite-volume discretization of the time-of-flight equation (13.1) in vector form as $\boldsymbol{V}\boldsymbol{\tau} = \boldsymbol{\phi}$. Here, $\boldsymbol{\tau}$ is the vector of unknown time-of-flight values, $\boldsymbol{\phi}$ is the vector of pore volumes per cell, and $\boldsymbol{V}$ is the upstream flux matrix corresponding to the operator $\vec{v}\cdot\nabla$. Consider a cell j with total influx v^i_j, and let $\boldsymbol{c}^j_b$ denote the vector of *backward* tracer concentrations corresponding to an imaginary experiment in which a tracer is injected in cell j and allowed to flow in the reverse direction of $\boldsymbol{v}$. Let $\boldsymbol{\chi}_j$ denote the characteristic vector, which equals one in cell j and zero in all other cells. We can then write the discrete equations for the forward time-of-flight $\boldsymbol{\tau}$ and the backward influence region $\boldsymbol{c}^j_b$ as follows

$$\boldsymbol{V}\boldsymbol{\tau}_f = \boldsymbol{\phi} \quad \text{and} \quad \boldsymbol{V}^T\boldsymbol{c}^j_b = \boldsymbol{\chi}_j v^i_j. \tag{13.13}$$

From this, we obtain the following expression for the time-of-flight value in cell j

$$\boldsymbol{\tau}_f[j] = \boldsymbol{\chi}^T_j \boldsymbol{V}^{-1}\boldsymbol{\phi} = \left(\frac{1}{v^i_j}\boldsymbol{c}^j_b\right)^T \boldsymbol{\phi}. \tag{13.14}$$

Hence, $\boldsymbol{\tau}_f[j]$ equals the pore volume of the backward influence region of cell j divided by the influx into the cell. If cell j lies close to a producer, the backward influence region typically contains flow paths with very different residence times. This implies that the forward time-of-flight value computed by the finite-volume method represents the average of a distribution that potentially can have (very) large variance. A similar argument applies to the backward time-of-flight. The averaging will introduce a systematic bias when we use averaged time-of-flight values to compute the dynamic heterogeneity measures discussed in the previous section. This bias may be acceptable in many applications, e.g., if we use dynamic heterogeneity measures to rank different model realizations or as a simple reduced-order model to predict recovery of secondary oil recovery [144, 218]. On the other hand, residence times computed from time-of-flight values will in most cases *overestimate* the time to breakthrough in heterogeneous displacements.

For a better description of the dynamic heterogeneity of a reservoir model, we can consider the *distribution* of time-of-flight for each cell. This is particularly interesting for cells perforated by production wells, since pointwise time-of-flight values in these cells describe the residence times, or time to breakthrough, for individual flow paths. To determine this *residence-time distribution* (RTD), we start by considering the linear transport equation

$$\phi\frac{\partial c}{\partial t} + \vec{v}\cdot\nabla c = 0, \quad c|_{\Gamma_i} = \delta(t), c(\vec{x},0) = 0, \tag{13.15}$$

which describes the transport of a unit pulse through a domain Ω from an inflow boundary Γ_i to an outflow boundary Γ_o. (The inflow and outflow will obviously be through wells in most reservoir models, but in the following I refer to these as boundaries for mathematical convenience.) For each point $\vec{x}$, the normalized time-of-flight distribution $\mathcal{T}(\cdot;\vec{x})$ is now simply defined by the Dirac function

$$\mathcal{T}(t;\vec{x}) = c(\vec{x},t) = \delta\big(t - \tau(\vec{x})\big). \tag{13.16}$$

At the outflow, the normalized RTD is given as

$$\mathcal{T}_o(t) = \frac{1}{F_o}\int_{\Gamma_o} c\,\vec{v}\cdot\vec{n}\,ds, \qquad F_o = \int_{\Gamma_o}\vec{v}\cdot\vec{n}\,ds. \tag{13.17}$$

It follows from the definition of the Dirac distribution that $\int \mathcal{T}_o(t)\,dt = 1$. From the RTD, we obtain flow capacity and storage capacity as follows [277]

$$F(t) = \int_0^t \mathcal{T}_o(s)\,ds, \qquad \Phi(t) = \frac{F_o}{\Phi_o}\int_0^t s\,\mathcal{T}_o(s)\,ds. \tag{13.18}$$

As before, both quantities are normalized so that $F(\infty) = \Phi(\infty) = 1$. From this definition, it also follows that the mean value of $\mathcal{T}_o(t)$ corresponds to the time $\bar{t} = \Phi_o/F_o$ it takes to inject one pore volume (1 PVI).

To compute discrete approximations to the time-of-flight and RTD, we introduce a finite-volume approximation,

$$\frac{d\boldsymbol{c}}{dt} + \boldsymbol{M}\boldsymbol{c} = 0, \quad \boldsymbol{c}(0) = \boldsymbol{c}_0 = \frac{\boldsymbol{q}_i}{\boldsymbol{\phi}}. \tag{13.19}$$

Here, $\boldsymbol{M}$ is the discretization of the linear operator $\phi^{-1}\vec{v}\cdot\nabla$ defined in the same way as for the time-of-flight equation, and $\boldsymbol{q}_i$ is the vector of inflow flow rates. In the `diagnostics` module, we use a backward Euler method to solve (13.19) directly. This method is robust and can take large time steps, but is not very accurate.

Alternatively, you can write the solution formally in terms of matrix exponentials $\boldsymbol{c}(t) = \exp(-t\boldsymbol{M})\boldsymbol{c}_0$. Let $\boldsymbol{q}_o$ be the vector that collects the flux across all outflow boundaries of individual cells, $\mathbf{1}$ be a vector with value one in each cell, and Φ_o be the total pore volume drained by the outflow boundary Γ_o. We then represent the discrete counterparts of (13.16)–(13.17) as follows

$$\boldsymbol{\mathcal{T}}(t)[j] = \boldsymbol{\chi}_j^T \exp(-t\boldsymbol{M})\boldsymbol{c}_0 \quad \text{and} \quad \mathcal{T}_o(t) = \boldsymbol{q}_o^T \exp(-t\boldsymbol{M})\boldsymbol{c}_0/\boldsymbol{q}_o^T\mathbf{1}. \tag{13.20}$$

The vector $\boldsymbol{\mathcal{T}}$ contains time-of-flight distributions in all cells (with value $\boldsymbol{\mathcal{T}}[j]$ in cell j) and $\mathcal{T}_o$ is the RTD over Γ_o. You can then employ a rational Padé approximation to evaluate the action of the matrix exponential,

$$\boldsymbol{\mathcal{T}}(t+\Delta t) = \exp(-\Delta t\boldsymbol{M})\boldsymbol{\mathcal{T}}(t) \approx P(-\Delta t\boldsymbol{M})\,Q(-\Delta t\boldsymbol{M})^{-1}\boldsymbol{\mathcal{T}}(t), \tag{13.21}$$

for suitable polynomials P and Q. One example would be to use the first-order polynomials $P(x) = 1 + x/2$ and $Q(x) = 1 - x/2$ to reduce fill in, i.e.,

$$\boldsymbol{\mathcal{T}}(t+\Delta t) = \big(\boldsymbol{I} - \tfrac{1}{2}\Delta t\,\boldsymbol{M}\big)\big(\boldsymbol{I} + \tfrac{1}{2}\Delta t\,\boldsymbol{M}\big)^{-1}\boldsymbol{\mathcal{T}}(t).$$

For each successive value you compute, you must solve a linear system and perform a matrix multiplication. The matrix $\boldsymbol{I} + \boldsymbol{M}$ has the same sparsity as the discretization matrix for the time-of-flight equation; i.e., is triangular, possibly after a permutation. Each linear solve is thus highly efficient, but the time step is restricted by a CFL condition because of the explicit part of the discretization coming from the polynomial $P(x)$. In our experience, using backward Euler is thus more efficient.

The next two examples illustrate the difference between averaged values and distributions and how these differences impact measures of dynamic heterogeneity.

Example 13.3.1 *We consider a single horizontal layer from SPE 10 with an injector along the south boundary and a producer along the north boundary. Assume we have computed a flow solution with fluxes represented in* `state`*, cell-wise pore volumes in* `pv`*, time-of-flight and tracer values represented in the structure* `D` *discussed on page 480, and well pair information in* `WP`*. The following routine computes the propagating pulse using (13.19) with a backward Euler time discretization and extracts the outflux observed in the producer as function of time:*

```
rtd = computeRTD(state, G, pv, D, WP, W);
```

The solid lines in Figure 13.7 show an unnormalized version of $\mathcal{T}_o(t)$ as function of time for two layers; here the integral equals the total allocation. The leading pulse for the Tarbert layer is spread out and has a small secondary hump. For Upper Ness, the pulse breaks through earlier and is more focused because of high-permeable channels connecting the

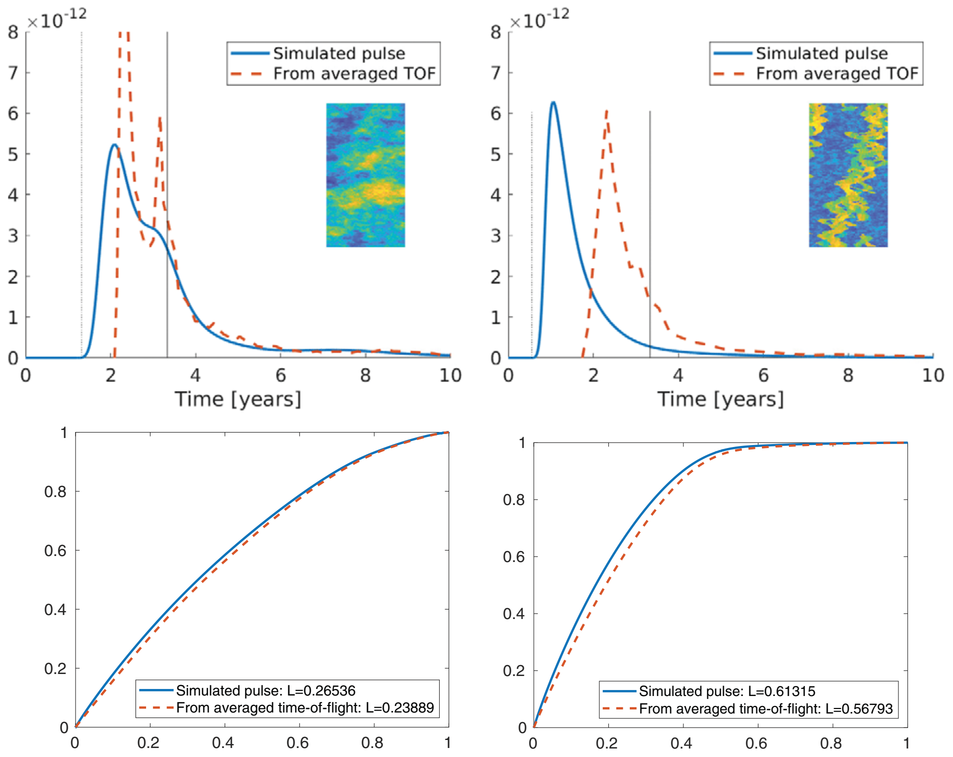

Figure 13.7 Residence-time distributions and F–Φ diagrams for two permeability field sampled from the SPE 10 benchmark. Solid lines are computed by tracing a unit pulse through the reservoir to determine the distribution at the outlet. Dashed lines are obtained by solving the time-of-flight equation by a finite-volume method and backing out data for representative flow paths through each of the cells of the model. Thin solid and dotted lines represent the mean of the distribution, i.e., the time it takes to inject one pore volume, and the time to breakthrough for the fastest flow path.

south and north boundaries. The mean of each distribution equals 1 PVI by construction. This may not be apparent from the plots since the distributions have very long tails, particularly in the channelized case.

We can also estimate this distribution from the average residence times $\boldsymbol{\tau}_r = \boldsymbol{\tau}_f + \boldsymbol{\tau}_b$, obtained by solving the forward and backward time-of-flight equations. We start by using the relationship $f_j = \boldsymbol{\phi}[j]/\boldsymbol{\tau}_r[j]$ to back out the flux

```
D  = computeTOFandTracer(state, G, rock, 'wells', W, ..)
f  = poreVolume(G,rock)./sum(D.tof,2);
```

Plotting $\boldsymbol{f}$ against $\boldsymbol{\tau}_r$ values sorted in ascending order and normalized by 1 PVI gives a discrete *(point) description of how a given total flux through the permeable medium will distribute itself over a finite number of flow paths. This distribution is highly irregular, and to get a* continuous *distribution we sort the $\boldsymbol{\tau}_r$ values into small bins, sum the total flux within this bin, and compute an averaged flux density value over the whole length of the bin. This is implemented in the function*

```
RTD = estimateRTD(pv, D, WP);
```

Using averaged time-of-flight values introduces a significant delay in the breakthrough time, in particular for Upper Ness. The two types of distributions also suffer from different types of numerical errors: Tracing a pulse by a finite-volume method preserves flux allocation and not pore volume and can also contain significant temporal smearing. Backing out a distribution from averaged time-of-flight values preserves total volume but not flux allocation. This is taken into account by the following function, which computes F–Φ diagrams from a RTD,

```
[f, phi] = computeFandPhi(rtd);
```

Figure 13.7 also compares the resulting F–Φ diagram with similar diagrams computed directly from τ_r and pore volume. As expected, the diagrams computed from the RTD are more concave and the corresponding Lorenz coefficients are somewhat larger than when computed from averaged time-of-flight values. However, the differences are not very large, and the measures computed from averaged time-of-flight values carry a systematic underestimation bias and can thus still be robustly used to rank and discriminate different cases.

Notice that we do *not* report F–Φ diagrams derived from *estimated* RTDs. The binning and averaging used in the estimation process introduces a strong smoothing that generally obfuscates dynamic heterogeneity measures. Hence, we strongly advise against computing any heterogeneity measure from *estimated* RTDs; these distributions should only be used for illustrations like Figure 13.7.

Example 13.3.2 *We repeat the same setup as in the previous example, but now with an injector placed slightly off-center along the south perimeter and two producers placed symmetrically at the northern perimeter. Figure 13.8 reports residence distributions and F–Φ diagrams for each of the two injector pairs. In both cases, producer P2 is connected*

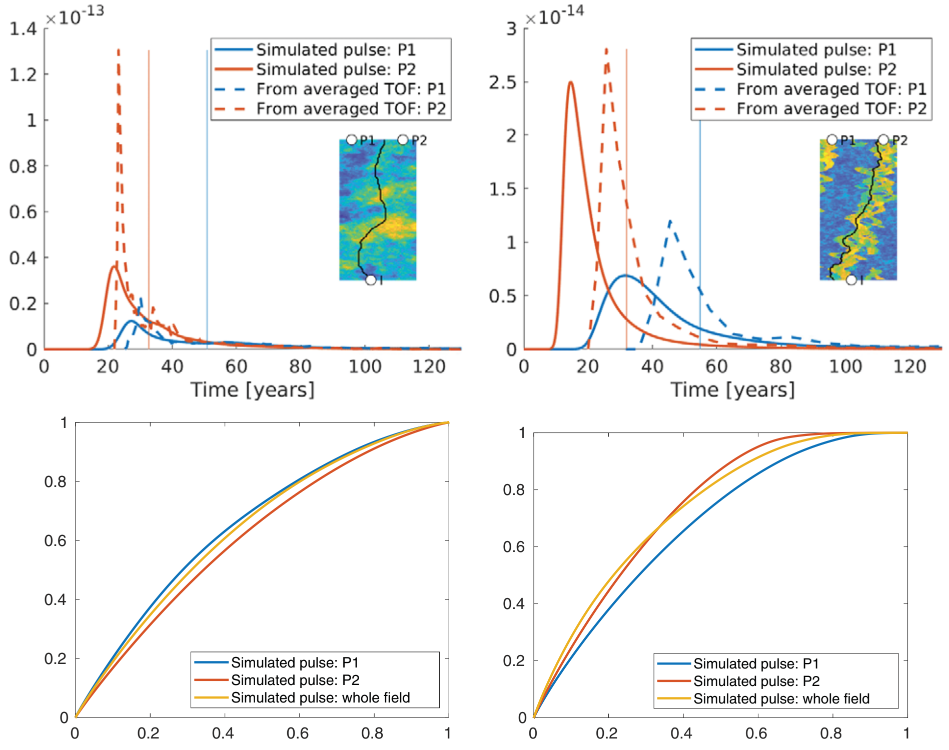

Figure 13.8 Residence-time distributions and F–Φ diagrams for two permeability field sampled from the SPE 10 benchmark with one injector and two producers.

to the injector through more permeable rocks than producer P1, even though the physical distance from I to P2 is larger than to P1. The fastest flow paths thus break through earlier in P2 than in P1, and overall, producer P2 allocates a significantly larger portion of the flux than P1.

Next, we look at the dynamic heterogeneity. For Tarbert, the (I,P2) well pair allocates approximately twice as much flux and a 30% larger flow region than (I,P1). The majority of the flow paths in the (I,P2) have small residence times and dynamically, this region is less heterogeneous than (I,P1), whose RTD contains a heavier tail. Hence, the F–Φ curve of P1 lies above P2 and has a higher Lorenz coefficient. For the reservoir as a whole, these two effects balance each other: In the beginning, the produced tracer comes predominantly from fast flow paths connected to P2, whereas the last tracer volumes produced come predominantly from slow flow paths connected to P1. Hence, fast flow paths from P2 ensure that the F–Φ curve of the whole reservoir lies below the curve of P1, whereas contributions from the heavier tail of P1 ensure that the overall curve lies above that of P2.

For Upper Ness, 50% more flux is allocated to P2 even though the associated flow region is smaller than for P1. In the (I,P1) region, all flow paths traverse regions of both high and

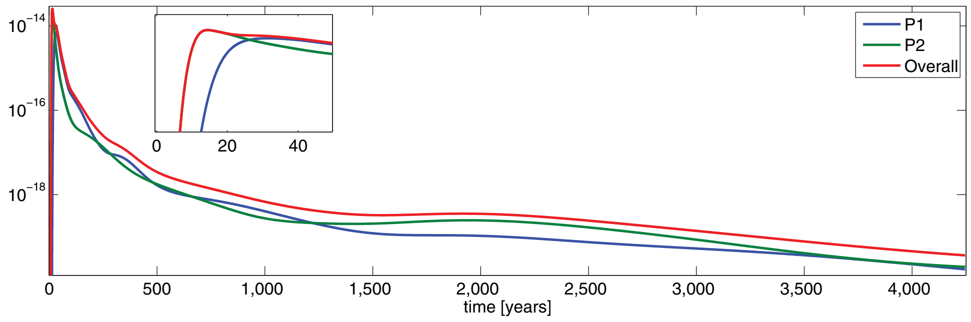

Figure 13.9 Residence-time distributions for the Upper Ness case shown on a semi-logarithmic axis. (One PVI corresponds to 41 years.)

low permeability, which means that this region is more homogeneous than the (I,P2) region, which contains a large fraction of flow paths that (almost) exclusively traverse cells with high permeability. Hence, the F–Φ curve of P2 lies above P1. Explaining the F–Φ curve of the whole reservoir is significantly more difficult. Figure 13.9 shows the RTD on a semi-logarithmic axis. During the first 20 years, tracer production is predominantly from P2. Producer P1 has the largest contribution from year 27 to year 1220, and then P2 contributes most for the rest of the simulated period. Obviously, this exercise has nothing to do with real reservoir engineering but serves to illustrate how tiny details on (unrealistically) long time scales affect measures like F–Φ diagrams.

Distributions of residence times are used in the study of chemical reactors and tracer tests [277, 139]. In [218] we demonstrate how you can convolve these distributions with representative, 1D displacement profiles to develop reduced-order models that, e.g., enable you to quickly estimate and discriminate improvements in macroscopic and microscopic displacement efficiency for polymer flooding.

13.4 Case Studies

The use of flow diagnostics is best explained through examples. In this section we therefore go through several cases and demonstrate various ways flow diagnostics can be used to enhance our understanding of flow patterns and volumetric connections, tell us how to change operational parameters such as well placement and well rates to improve recovery, and so on.

13.4.1 Tarbert Formation: Volumetric Connections

As our first example, we consider a subset of the SPE 10 data set consisting of the top twenty layers of the Tarbert formation; see Section 2.5.3. We modify the original inverted five-spot well pattern by replacing the central injector by two injectors that are moved a

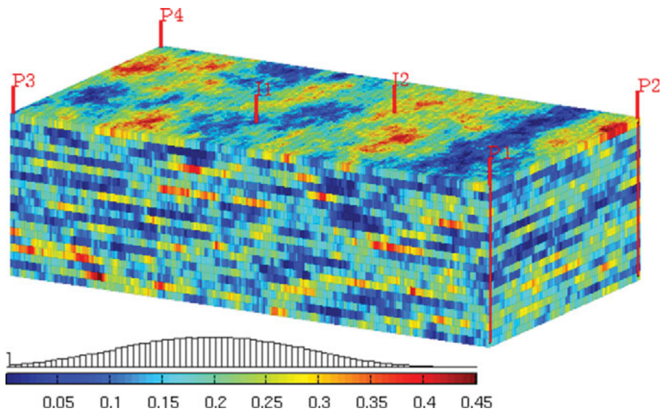

Figure 13.10 Porosity and well positions for a model consisting of subset of the Tarbert formation in Model 2 from the 10th SPE Comparative Solution Project

short distance from the model center (see Figure 13.10), assume single-phase incompressible flow, and solve the corresponding flow problem. You can find complete description of the setup in `showWellPairsSPE10.m` in the `book` module.

Given the geological model represented in terms of the structures `G` and `rock`, the wells represented by `W`, and the reservoir state including fluxes by `rS`, we first compute the time-of-flight and volumetric partitions:

```
D = computeTOFandTracer(rS, G, rock, 'wells', W);
```

This gives us the information we need to partition the volume into different drainage and sweep volumes. The simplest way to do this for the purpose of visualization is to use a majority vote over the injector and producer partitions to determine the well that influences each cell the most as shown in Figure 13.11. The result of this majority vote is collected in `D.ppart` and `D.ipart`, respectively, and the essential commands to produce the two upper plots in Figure 13.11 are:

```
plotCellData(G,D.ipart, ..);
plotCellData(G,D.ppart,D.ppart>1, ..);
```

Since there are two injectors, we would expect to only see two colors in the to plot of sweep regions. However, there is also a small blue volume inside the triangular section bounded by I1, P1, and P2, which corresponds to an almost impermeable part of the reservoir that will not be swept by any of the injectors. The well pattern is symmetric and for a homogeneous medium we would therefore expect that the two pressure-controlled injectors would sweep symmetric volumes of equal size. However, the Tarbert formation is highly heterogeneous and hence the sweep regions are irregular and clearly not symmetric. The injection rate of I2 is approximately six times that of I1 because the wells are completed in cells with very different permeability, hence I2 will sweep a much larger region than I1. In particular, we

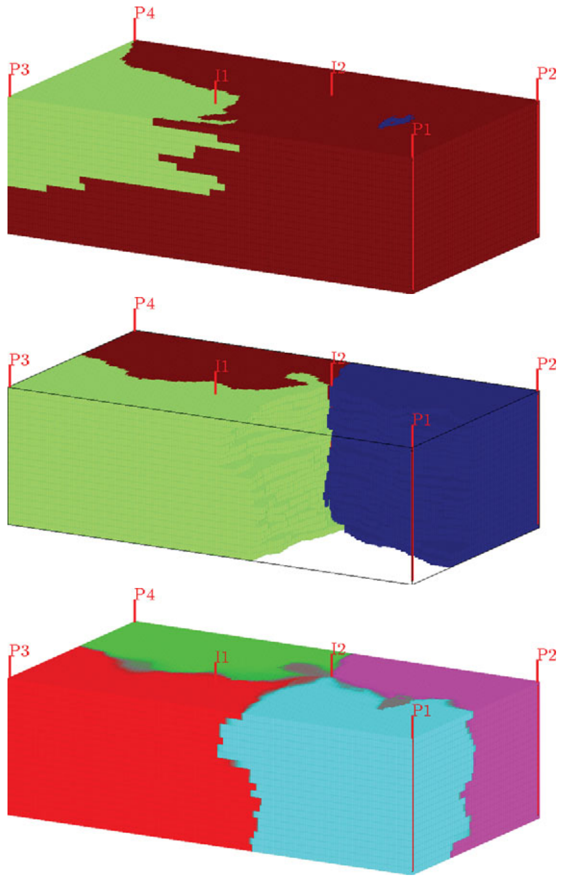

Figure 13.11 Sweep (top) and drainage regions (middle) determined by a majority vote over injector and producer partitions, respectively, for the Tarbert model. The bottom plot shows producer influence region without majority vote, with gray color signifying cells that affect multiple producers.

see that I2 is the injector that contributes most to flooding the lower parts of the region near P3, even though I1 is located closer. Looking at the drainage and sweep regions in conjunction, it does not seem likely that I1 will contribute significantly to support the production from wells P1 and P2 unless we increase its rate.

When using a majority vote to determine drainage and sweep regions, we disregard that individual cells may be influenced by more than one well. To better visualize the influence regions, we can blend in a gray color in all cells in which more than one color has nonzero concentration, as shown in the bottom plot of Figure 13.11. The plot is generated by the following call:

```
plotTracerBlend(G, D.ppart, max(D.ptracer, [], 2), ..);
```

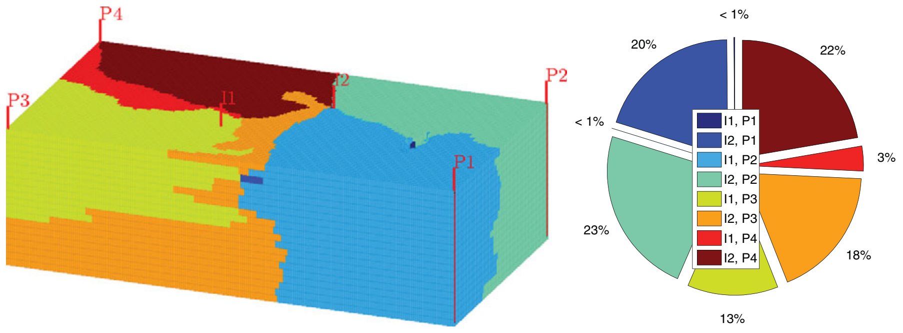

Figure 13.12 Well-pair regions and associated fraction of the total pore volume for the upper 20 layers of the Tarbert formation.

Having established the injection and producer partitions, we can identify well pairs and compute the pore volumes of the region associated with each pair:

```
WP = computeWellPairs(rS, G, rock, W, D);
pie(WP.vols, ones(size(WP.vols)));
legend(WP.pairs,'location','Best');
```

To visualize the volumetric regions, we compute the tensor product of the injector and producer partitions and then compress the result to get a contiguous partition vector with a zero value signifying unswept regions:

```
p = compressPartition(D.ipart + D.ppart*max(D.ipart))-1;
plotCellData(G,p,p>0,'EdgeColor','k','EdgeAlpha',.05);
```

The result shown in Figure 13.12 confirms our previous observations of the relative importance of I1 and I2. Altogether, I1 contributes to sweep approximately 16% of the total pore volume, shown as the light red and the yellow regions in the 3D plot.

It is also interesting to see how these volumetric connections affect the fluxes in and out of wells. To this end, we should look at the cumulative well allocation factors, which are defined as the cumulative flux in/out of a well from the bottom to the top perforation of a vertical well, or from toe to heel for a deviated well. We start by computing the flux allocation manually for the two injectors (outflow fractions are given from well head and downward and hence must be flipped):

```
for i=1:numel(D.inj)
   subplot(1,numel(D.inj),i); title(W(D.inj(i)).name);
   alloc = cumsum(flipud(WP.inj(i)).alloc,1);
   barh(flipud(WP.inj(i).z), alloc,'stacked'); axis tight
   lh = legend(W(D.prod).name,4);
   set(gca,'YDir','reverse');
end
```

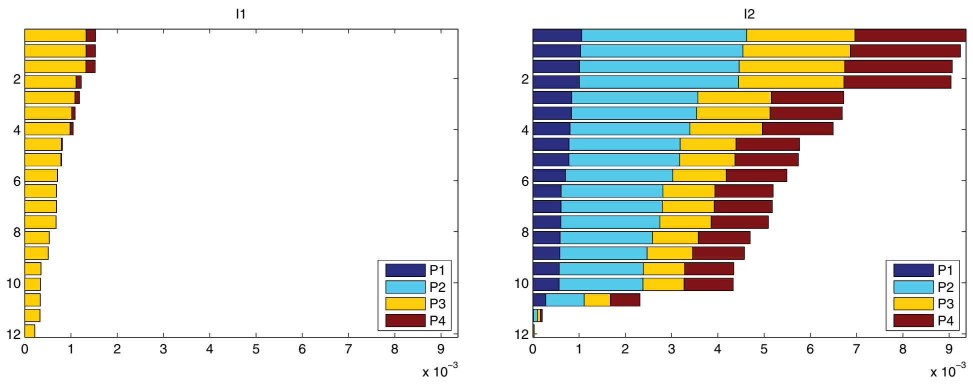

Figure 13.13 Well allocation factors for the two injectors of the Tarbert model.

Figure 13.13 shows the resulting bar plots of the cumulative allocation factors. These plots confirm and extend the understanding we have developed by studying volumetric connections: I1 will primarily push fluids towards P3. Some fluids are also pushed towards P4, and we observe that there is almost no outflow in the top three perforations where the rock has low quality. Injection from I2, on the other hand, contributes to uphold the flux into all four producers. We also see that the overall flux is not well balanced. Producer P1 has significantly lower inflow than the other three. Alternatively, we can use `plotWellAllocationPanel(D, WP)` to compute and visualize the well allocation factors for all the wells in the model, as shown in Figure 13.14. (This function is only useful for a small number of wells.)

When your simulation case contains only a few well pairs, you may get significantly more information about the flow patterns in the reservoir if you subdivide the wells into multiple segments (sets of completions). To exemplify, let us look more closely at the performance of the different completions along the well paths. To this end, we divide the completion intervals into bins and assign a corresponding set of pseudo-wells for which we recompute flow diagnostics. As an example, we split the completions of I1 into three bins and the completions of I2 into four bins.

```
[rSp,Wp] = expandWellCompletions(rS,W,[5, 3; 6, 4]);
Dp = computeTOFandTracer(rSp, G, rock, 'wells', Wp);
```

Figure 13.15 shows the majority-voted sweep regions for the four segments of I2. To better see the various sweep regions, the 3D plot is rotated 180 degrees compared with the other 3D plots of this model. To obtain the figure, we used the following key statements:

```
plotCellData(G, Dp.ipart,(Dp.ipart>3) & (Dp.ipart<8),...);
WPp = computeWellPairs(rSp, G, rock, Wp, Dp);
avols = accumarray(WPp.pairIx(:,1),WPp.vols);
pie(avols(4:end));
```

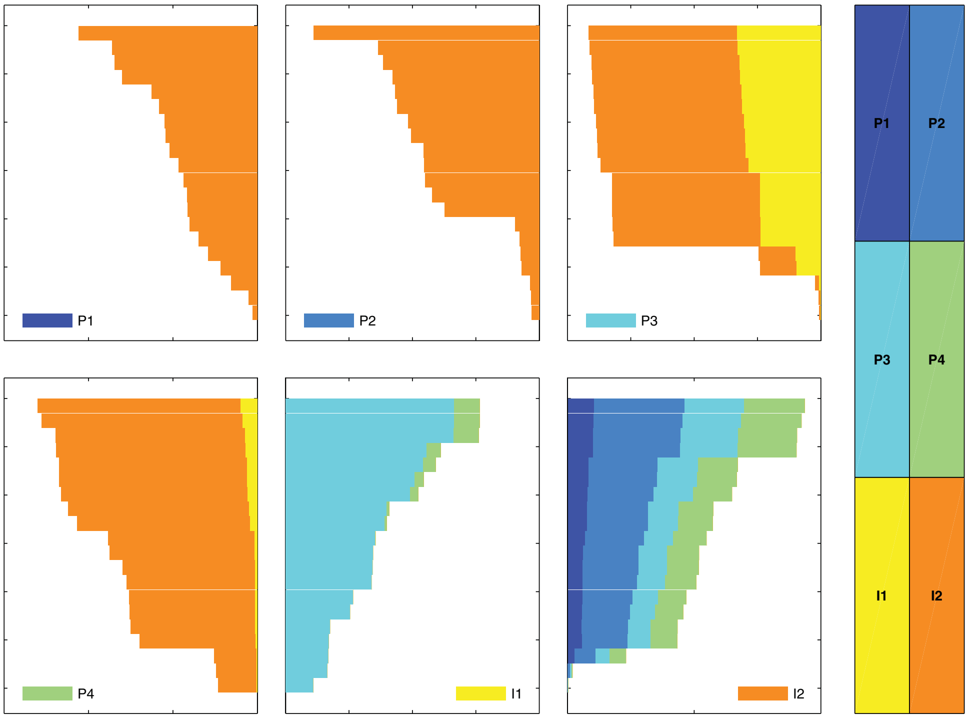

Figure 13.14 Normalized well allocation factors for all wells of the Tarbert model.

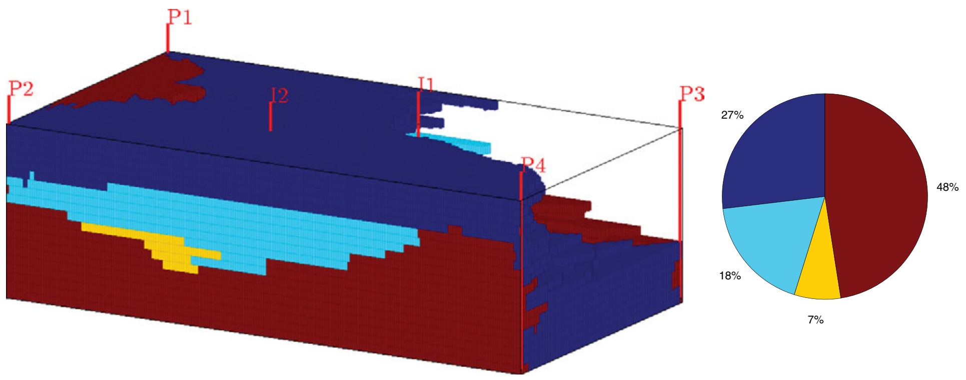

Figure 13.15 Majority-voted sweep regions for the Tarbert case with I2 of divided into four segments that each are completed in five layers of the model. (Notice that the view angle is rotated 180 degrees compared with Figure 13.11.) The pie chart shows the fraction of the total sweep region that can attributed to each segment of the well.

Notice, in particular, that fluids injected in the lowest segment is the major contributor in almost half of the well's total sweep region.

13.4.2 Heterogeneity and Optimized Well Placement

In this example, we first compute the Lorenz coefficient for all layers of the SPE 10 model subject to an inverted five-spot well pattern. We then pick one of the layers and show how we can balance the well allocation and improve the Lorenz coefficient and the areal sweep by moving some of the wells to regions with better sand quality.

To compute Lorenz coefficient for all layers in the SPE 10 model, we first define a suitable $60 \times 220 \times 1$ grid covering a rectangular area of $1{,}200 \times 2{,}200 \times 2$ ft^3. Then, we loop over all the 85 layers using the following essential lines:

```
for n=1:85
   rock = getSPE10rock(1:cartDims(1),1:cartDims(2),n);
   rock.poro = max(rock.poro, 1e-4);
   T  = computeTrans(G, rock);

   for w = 1:numel(wtype),
      W = verticalWell(..);
   end
   rS = incompTPFA(initState(G, W, 0), G, T, fluid, 'wells', W);

   D       = computeTOFandTracer(rS, G, rock, 'wells', W, 'maxTOF', inf);
   [F,Phi] = computeFandPhi(poreVolume(G,rock), D.tof);
   Lc(n)   = computeLorenz(F,Phi);
end
```

Transmissibility is recomputed inside the loop because the permeability changes for each new layer. Likewise, we regenerate the well objects to ensure correct well indices when updating the petrophysical data. You find the complete source code in the script `computeLorenzSPE10` in the `book` module.

Figure 13.16 reports Lorenz coefficients for all layers when the injector and the four producers are controlled by bottom-hole pressure. The relatively large dynamic heterogeneity and the variation among individual layers within the same formation is a result of our choice of well controls. The actual injection and production rates achieved with pressure-controlled wells are very sensitive to each well being perforated in a region of good sand quality. This is almost impossible to ensure when using fixed well positions for all layers of SPE 10, and hence pressure-controlled wells will generally accentuate heterogeneity effects. To see this, you should modify the script and rerun the case with equal rates in all four producers.

Our overall setup and somewhat haphazard well placement is not representative of a real reservoir, but serves well to illustrate our next point. Since the Lorenz coefficient generally is quite large, most of the cases would have suffered from early breakthrough if we were

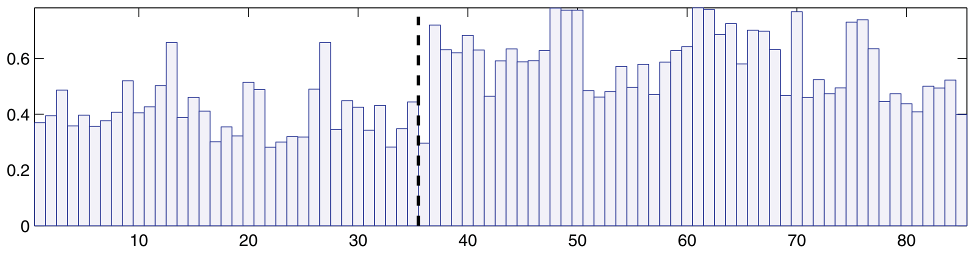

Figure 13.16 Lorenz coefficient for the 85 horizontal layers of the SPE 10 model subject to an inverted five-spot pattern with injector and producers controlled by bottom-hole pressure. The dashed line shows the border between the Tarbert and Upper Ness layers.

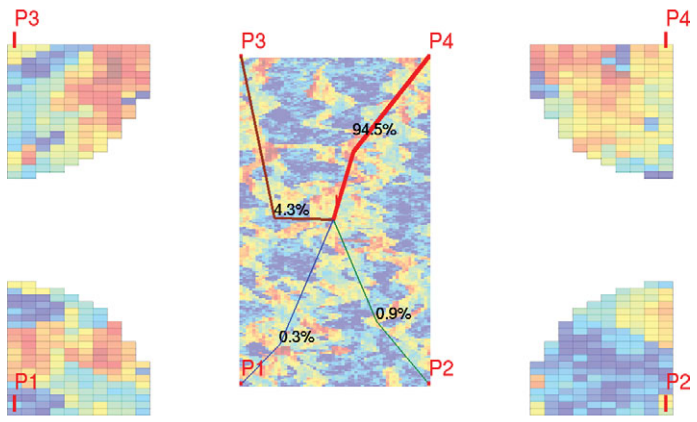

Figure 13.17 Well configuration and flux allocation for the four well pairs with initial well configuration for Layer 61. (Red colors are good sands, while blue colors signify sands of low permeability and porosity.)

to use this initial well placement for multiphase fluid displacement. Let us therefore pick one of the layers and see if we can try to improve the Lorenz coefficient and hence also the sweep efficiency. Figure 13.17 shows the well allocation and the sand quality near the producers for Layer 61, which is the layer giving the worst Lorenz coefficient. The flux allocation shows that we have a very unbalanced displacement pattern where producer P4 draws 94.5% of the flux from the injector and producers P1 and P2 together only draw 1.2%. This can be explained by looking at the sand quality in our reservoir. Producer P1 is completed in a low-quality sand and will therefore achieve low injectivity if all producers operate at the same bottom-hole pressure. Producers P2 and P3 are perforated in cells with better sand, but are both completely encapsulated in regions of low sand quality. On the other hand, producer P4 is connected to the injector through a relatively contiguous region of high-permeable sand.

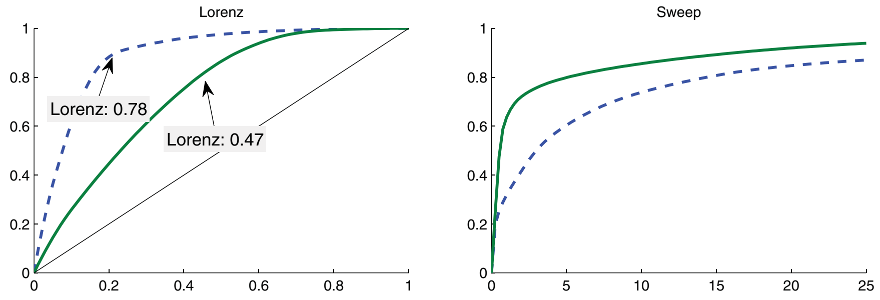

Figure 13.18 F–Φ and sweep diagrams before (dashed line) and after (solid line) producers P1 to P4 have been been moved to regions with better sand quality.

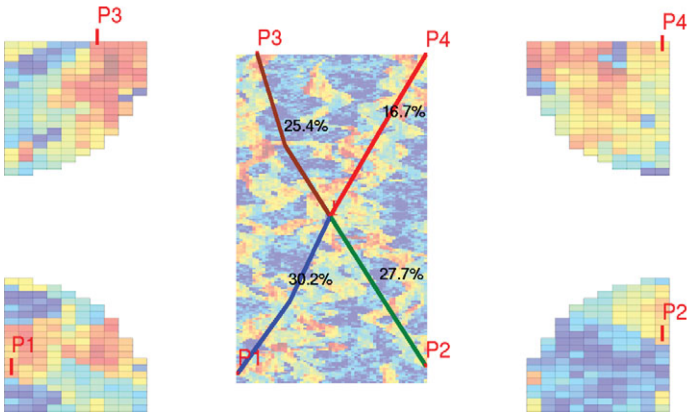

Figure 13.19 Well configuration and flux allocation for the four well pairs after all producers have been moved to regions with better sand quality.

The largely concave F–Φ diagram shown to the left in Figure 13.18 testifies that the displacement is strongly heterogeneous and is characterized by large differences in the residence time of different flow paths, or in other words, suffers from early breakthrough of the displacing fluid. Hence, we must inject large amounts of the displacing fluid to recover the hydrocarbons from the low-quality sand; this can be seen from the weakly concave sweep diagram to the right in Figure 13.18. Altogether, we should expect a very unfavorable volumetric sweep from this well placement.

To improve the displacement, we can try to move each producer to a better location. To this end, we should look for cells in the vicinity of each well that have better sand quality (higher porosity and permeability) *and* are connected to the injector by more contiguous paths of good quality sands. Figure 13.19 shows the well allocation after we have moved P1 to P3. Producer P4 was already in a good location and is only shifted one cell. The result is a much more balanced well allocation, which is confirmed by the rates reported in Table 13.1. (Notice also that the overall reservoir rate has increased slightly.) Figure 13.18

Table 13.1 *Volumetric flow rates in units* 10^{-5} m^3/s *for each of the wells in Layer 61 of the SPE 10 model. (These are obviously far from rates seen in real reservoirs.)*

Placement	P1	P2	P3	P4	I
Initial	−0.0007	−0.0020	−0.0102	−0.2219	0.2348
Improved	−0.0719	−0.0660	−0.0604	−0.0398	0.2381

shows how the F–Φ diagram has become significantly less concave and that the sweep diagram has become more concave. This testifies that the variation in residence times associated with different flow paths is much smaller. We should thus expect a more efficient and less heterogeneous volumetric sweep.

Dynamic heterogeneity measures correlate surprisingly well with recovery for waterflooding processes [278, 144, 259, 260, 218], and are generally easy to use as a guide when searching for optimal or improved well placement. Because of their low computational cost, the measures can be used as part of a manual, interactive search or combined with more rigorous mathematical optimization techniques. In the `diagnostics` module you can find examples that use flow diagnostics in combination with adjoint methods to determine optimal well locations and set optimal injection and production rates. The interested reader can find more details about optimization based on flow diagnostics in [218].

Computer exercises

13.4.1 Repeat the experiment using fixed rates for the producers (and possibly also for the injector). Can you explain the differences that you observe?

13.4.2 Compute Lorenz coefficients from RTDs instead and compare.

13.4.3 Use the interactive diagnostic tool introduced in the next section to manually adjust the bottom-hole pressures (or alternatively the production rates) to see if you can improve the sweep even further.

13.4.4 Can you devise an automated strategy that uses flow diagnostics to search for optimal five-spot patterns?

13.4.5 Cell-averaged time-of-flight values computed by a finite-volume scheme can be quite inaccurate when interpreted pointwise, particularly for highly heterogeneous media. To investigate this, pick one of the fluvial layers from SPE 10 and use the `pollock` routine from the `streamline` module to compute time-of-flight values on a 10×10 subsample inside a few selected cells. Alternatively, you can use the finite-volume method on a refined grid.

13.5 Interactive Flow Diagnostics Tools

The ideas behind most flow diagnostics techniques are relatively simple to describe and their computation is straightforward to implement. The real strength of these techniques nevertheless lies in their visual appeal and the ability for rapid user interaction. Together, MATLAB and MRST provide a wide variety of powerful visualization routines you can use to visualize input parameters and simulation results. The `diagnostics` module supplies additional tools for enhanced visualization, but using a script-based approach to visualization means that you each time need to write extra code lines to manually set color map and view angle or display various additional information such as legends, colorbars, wells, and figure titles.

We have mostly omitted these extra code lines in our previous discussion, but if you go in and examine the accompanying scripts, you will see that a large fraction of the code lines focus on improving the visual appearance of plots. Such code is repetitive and should ideally not be exposed to the user of flow diagnostics. More importantly, a script-based approach gives a static view of the data and offers limited capabilities for user interaction, apart from zooming, rotating, and moving the displayed data sets. Likewise, a new script must be written and executed each time we want to look a new plot that combines various types of diagnostic data, e.g., to visualize time-of-flight or petrophysical values within a given color region, use time-of-flight to threshold the color regions, etc.

Preprocessing: Predicting Fluid Movement

To simplify the user interface to flow diagnostics, we have integrated most of the flow diagnostics capabilities discussed earlier in the chapter into a graphical tool that enables you to interact more directly with your data set

```
interactiveDiagnostics(G, rock, W);
```

The interesting part of this GUI discussion is not the exact layout of the user interface, which likely will change in future versions of MRST, but the different quantities it can produce.

The GUI has primarily been written for preprocessing to predict future fluid movement and uses the standard two-point incompressible flow solver for a single-phase fluid with density 1,000 kg/m^3 and viscosity 1 cP to compute a representative flow field, which is then fed to the function `computeTOFandTracer` to compute the basic flow diagnostics quantities. Once the computation is complete, the graphical user interface is launched; see Figure 13.20: At the time of writing, this consists of a *plotting window* showing the reservoir model and a *control window*, which contains a set workflow tabs and menus that enable you to use flow diagnostics to explore volumetric connections, flow paths, and dynamic heterogeneity measures.

The control window has three different tabs. The “Region selection” tab is devoted to displaying the various kinds of volumetric regions discussed in Section 13.1.1. The “Plots” tab lets you compute F–Φ diagram, Lorenz coefficient, and the well allocation

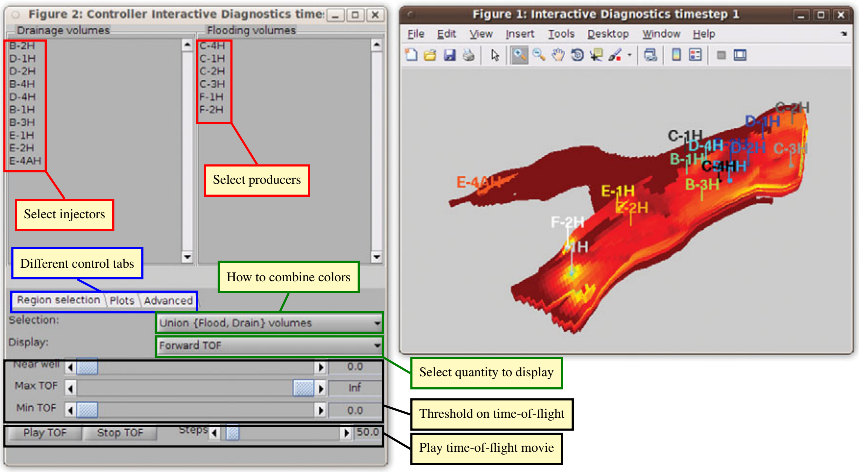

Figure 13.20 A graphical user interface to flow diagnostics. The plotting window shows forward time-of-flight, which is the default value displayed upon startup. In the control window, we show the "Region selection" tab that lets you select which quantity to show in the plotting window, select which wells to include in the plot, specify how to combine color concentrations to select volumetric regions, and as well set maximum and minimum time-of-flight values to crop the volumetric regions.

panels shown in Figure 13.14. From this tab, you can also bring up a dialog box to edit the well settings and recompute a new flow field and the resulting flow diagnostics (unless the parameter `'computeFlux'` is set to false). In the "Advanced" tab, you can control the appearance of the 3D plot by selecting whether to display grid lines, well-pair information, and well paths, set 3D lighting and transparency value, etc. This tab also allows you to export the current volumetric subset as a vector of boolean cell indicators.

Regions to be displayed in the plotting window are specified by selecting a set of active wells and by choosing how the corresponding colorings should be combined. That is, for a given set of active wells, you can either display all cells that have nonzero concentration for the injector *or* producer colors, only those cells that have nonzero concentration for injector *and* producer colors, only cells with nonzero concentration of producer colors, or only cells with nonzero concentrations for injector colors. To select a set of active wells, you can either use the list of injectors/producers in the control window, or mark wells in the plotting window. In the latter case, the set of active wells consist of the well you selected plus all other wells this well communicates with. Selecting a well in the 3D view also brings up a new window that displays a pie chart of the well allocation factors and a graph that displays cumulative allocation factors per connection. Figure 13.21 shows this type of visualization for a model where we also have access to a set of states from a full multiphase simulation. In this case, we invoke the GUI by calling

```
interactiveDiagnostics(G, rock, W, 'state', state, 'computeFlux', false);
```

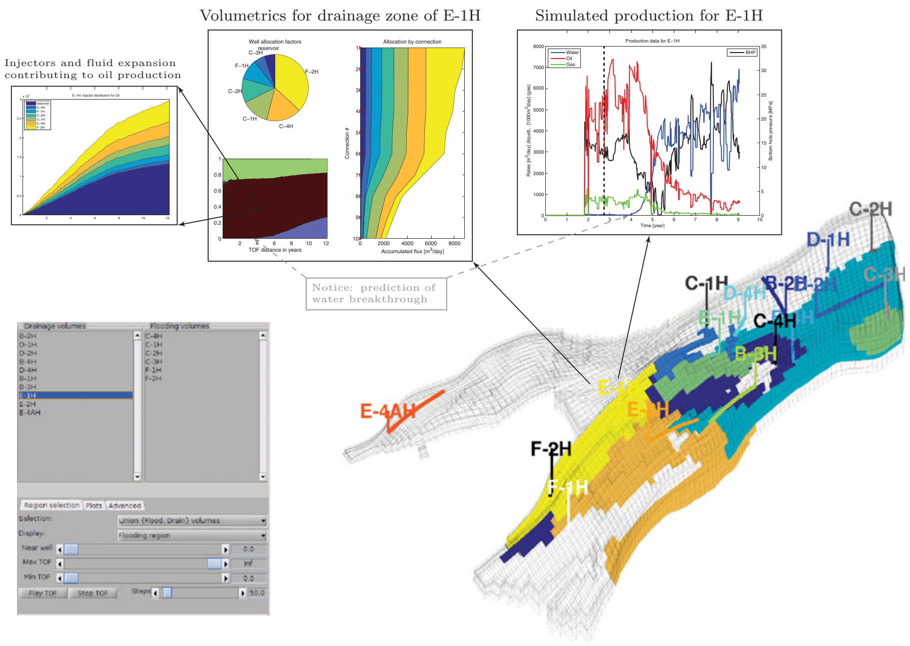

Figure 13.21 Flow diagnostics used to analyze volumetric connections and predict future inflow of fluids to well E-1H in a three-phase, black-oil flow simulation of the Norne field. For comparison, the figure also reports the *simulated* production from E-1H, where the dashed vertical line represent present time.

so that the fluxes used to compute time-of-flight and color concentrations are extracted from the given reservoir state given in `state`. Left-clicking on producer E-1H brings up a plot of the well allocation together with a plot of the fluid distribution displayed as function of the backward time-of-flight from the well. To further investigate the flow mechanism, we can mark each fluid and get a plot of how various injectors and fluid expansion in the reservoir contribute to push the given fluid toward the producer. In Figure 13.21 we have expanded the GUI by another function that plots the simulated well history of each well. This history confirms what we see in the backward time-of-flight analysis of the drainage zone: a few years into the future, water arrives at the producer (blue line), and there is a marked decay in oil and gas production (red and green lines).

The GUI also offers functionality to load additional cell-based data sets that can be displayed in the 3D plot:

```
interactiveDiagnostics(G, rock, W, celldata);
interactiveDiagnostics(G, rock, W, celldata, 'state', state);
```

and has some functionality for postprocessing simulations with multiple time steps, but this is beyond the scope of the current presentation.

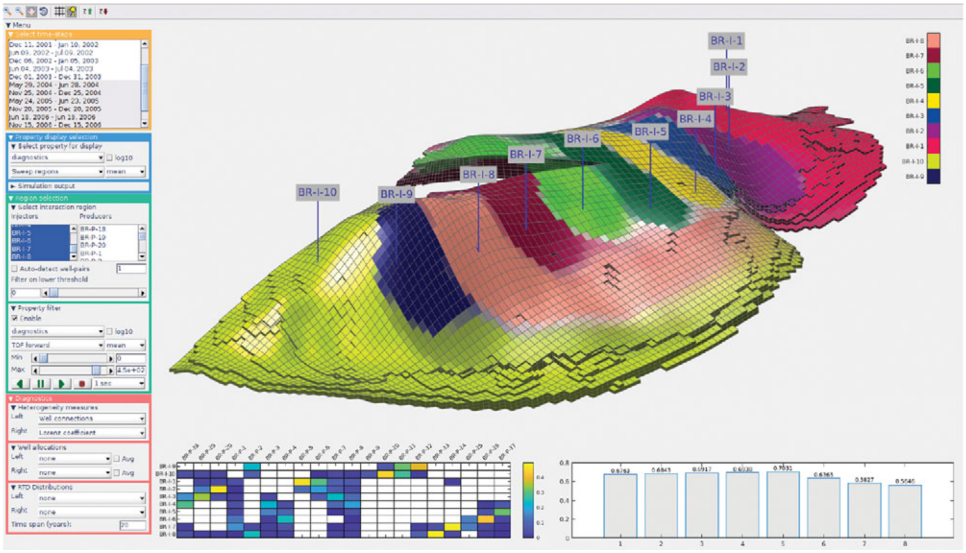

Figure 13.22 Variation in sweep regions over eight time report steps from an ECLIPSE simulation of the Brugge benchmark case. Clear color indicate regions swept by a single injector in all time steps, whereas grayish colors indicate regions being swept by more than one injector over the time interval.

Postprocessing: Enhanced Visualization of Simulation Results

As of release 2018b, MRST also offers another user interface dedicated entirely to *postprocessing* simulation data

```
PostProcessDiagnostics()
PostProcessDiagnostics('filename')
```

Figures 13.22–13.24 illustrate how flow diagnostics is used to enhance the understanding of the fluid communication for a full ECLIPSE simulation of the Brugge benchmark case [249].

Figure 13.22 shows the average sweep regions over eight report steps for all injectors, limited outward by an upper bound on the forward time-of-flight. Regions swept by a single injector over the whole simulation period are shown as clear colors, whereas diffuse or grayish colors represent regions swept by multiple injectors. You can use such plots to analyze how sweep or drainage regions change over time. The matrix plot to the lower left reports communication strength between each well pair, with rows representing injectors and columns injectors. The lower right plot shows how the dynamic heterogeneity measured by the Lorenz coefficient changes over the eight time steps. By selecting a single injector or producer, you can break up this analysis to see the dynamic heterogeneity of individual sweep, drainage, or well-pair regions, measured in terms of F–Φ diagrams or Lorenz coefficients.

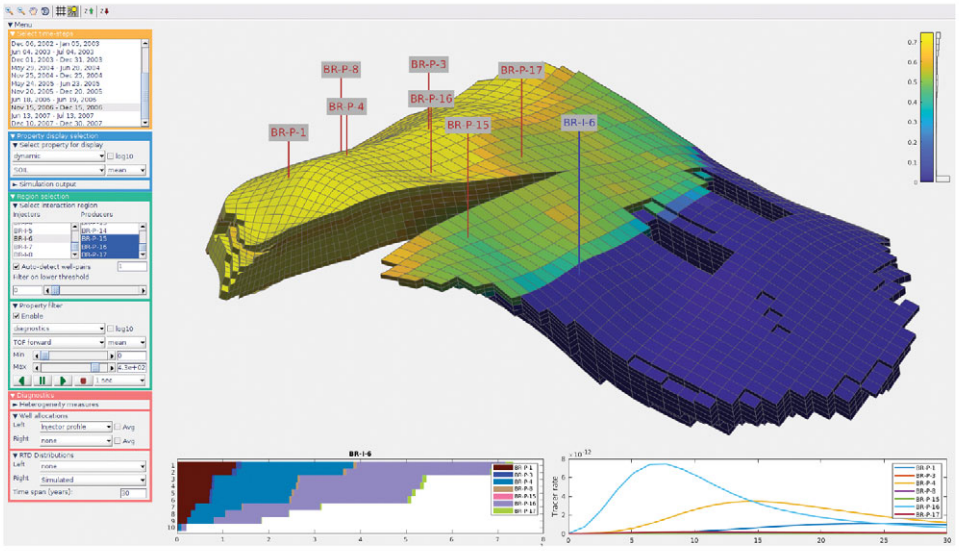

Figure 13.23 Sweep region of injector BR-I-6 for the Brugge benchmark case. The lower plots show flux allocation for the injector and the RTD for a passive tracer tracer released from the injector.

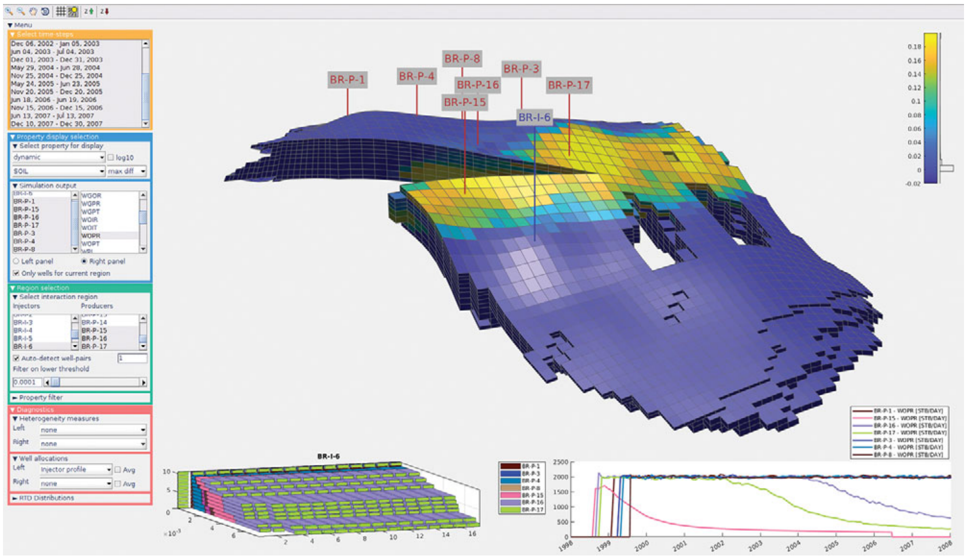

Figure 13.24 Saturation changes within the sweep region of injector BR-I-6 for the Brugge benchmark case. The lower-right plot reports flux allocation every half year for the injector from toe to heel (top to bottom in the plot). The lower-right plot reports water production rates for all connected producers.

Figure 13.23 shows oil saturation inside the sweep region of BR-I-6. From the flux allocation in the lower-left plot, we see that this injector primarily supports BR-P-16, BR-P-4, and BR-P-1 with the current set of drive mechanisms. Producer BR-P-15 lies closer than any other wells, but has been shut in (see lower-right plot in Figure 13.24) and is only shown since significant fluid volumes have been going to this producer earlier in the simulation. Also BR-P-17 lies closer to BR-I-6 than the other producers with large flux allocation, but as you can see from Figure 13.22, this well lies in the middle of BR-I-5's sweep region and is predominantly supported by that injector. The lower-right plot reports the first 30 years of an RTD analysis and shows that a passive tracer injected now from BR-I-6 will first break through in BR-P-16, and then in BR-P-4 and BR-P-1. This gives time lines for displacement fronts originating from BR-I-6.

Figure 13.24 reports accumulated saturation changes inside the sweep region over the whole simulation period. We clearly see how the displacement profile has engulfed producers BR-P-15 to 17 and caused a decline in production rates. In particular, BR-P-15 hardly managed to stay on plateu and started its decline less than half a year after it started operating.

The types of flow diagnostics discussed previously are also available in ResInsight, which is an open-source tool for postprocessing of simulation results on ECLIPSE format. This software is part of the OPM Initiative and currently functions as an outlet of research ideas from MRST towards professional use in industry.

In the rest of the chapter, we use the preprocessing GUI to study two reservoir models. The accompanying scripts do not reproduce all figures directly, but specify the manual actions you need to reproduce many of the figures.

13.5.1 Synthetic 2D Example: Improving Areal Sweep

The first example considers a slightly modified version of the setup from Figure 13.1. We have introduced two low-permeable barriers, moved the two injectors slightly to the south, and switched all wells from rate to pressure control; see Figure 13.25. Table 13.2 reports specific values for the well controls under the label "base case."

Our previous setup had a relatively symmetric well pattern in which producers P1 and P3 were supported by injectors I1 and I2, respectively, whereas producer P2 was supported by both injectors. This symmetry is broken by the two barriers, and now injector I1 also provides significant support for producer P3. You can infer this from the plot of producer partition: the gray areas between the magenta (P2) and red (P3) regions signify parts of the reservoir that are drained by both producers. The relatively large gray area southeast of I1 indicates that this injector supports both P2 and P3. There is also a gray area southwest of injector I2, but since this is less pronounced, we should only expect that only a small portion of P2's inflow can be attributed to I2. To confirm this, you can load the model in the interactive viewer (see the script `interactiveSimple`), and click the names for each individual producer to bring up a pie chart reporting the corresponding flux allocation.

Table 13.2 *Well controls given in terms of bottom-hole pressure [bar] for a simple 2D reservoir with five initial wells (I1, I2, P1, P2, P3) and one infill well (I3).*

	I1	I2	I3	P1	P2	P3	Lorenz
Base case	200	200	—	100	100	100	0.2730
Case 1	200	200	—	100	130	80	0.2234
Case 2	200	200	—	100	130	80	0.1934
Case 3	200	220	140	100	130	80	0.1887

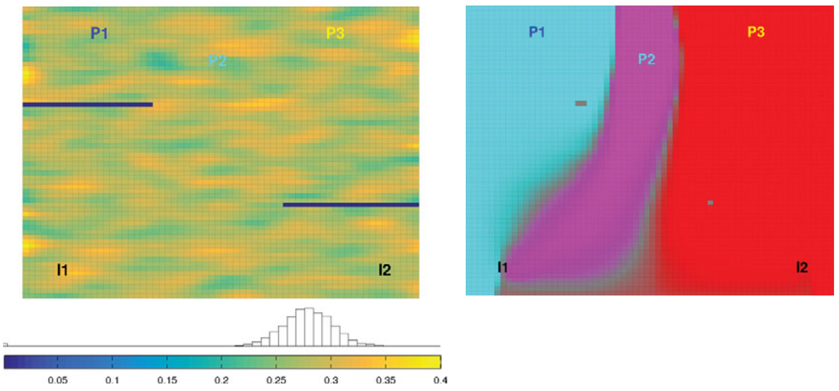

Figure 13.25 A simple 2D reservoir with two injectors and three producers. The left plot shows porosity and the right plot the corresponding producer partition.

To figure out to what extent this is a good well pattern or not, we start by looking at how a displacement front would propagate if the present flow field remains constant. For a displacement front traveling with a unit speed relatively to the Darcy velocity given by the flux field, the region swept by the front at time t consists of all cell for which $\tau_f \leq t$. Using the interactive GUI shown in Figure 13.20, you can either show swept regions by specifying threshold values manually, or use the “Play TOF” button to animate how the displacement front advances through the reservoir. Figure 13.26 shows four snapshots of such an advancing front. We notice, in particular, how the sealing fault to the northwest of the reservoir impedes the northbound propagation of the displacement front from I1 and leads to early breakthrough in P2. There is also a relatively large region that remains unswept after the two displacement fronts have broken through in all three producers.

As a more direct alternative to studying snapshots of an imaginary displacement front, we can plot the residence time, i.e., the sum of the forward and backward time-of-flight, as shown in the upper-left plot of Figure 13.27. Here, we use a nonlinear gray-map to more clearly distinguish high-flow zones (dark gray) from stagnant regions (white) and other

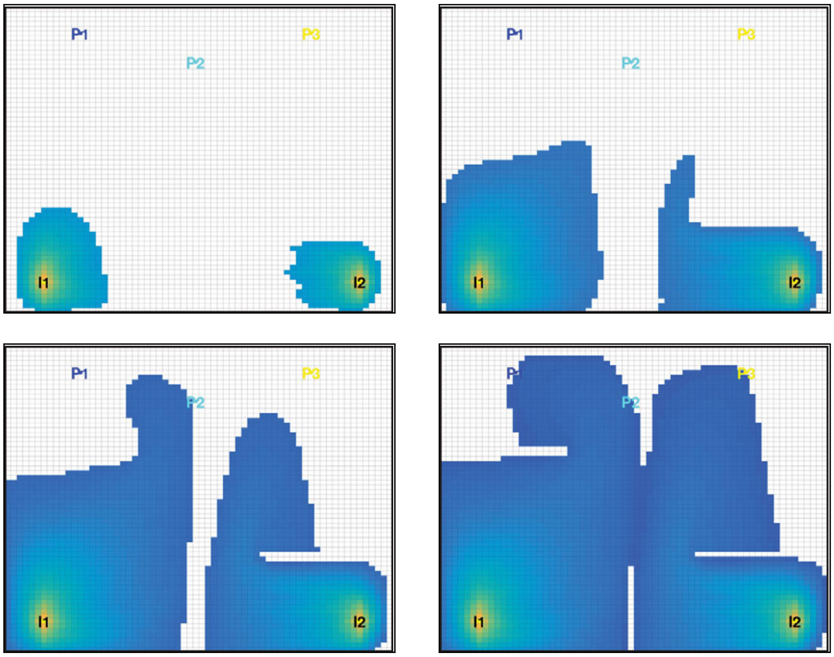

Figure 13.26 Evolution of an imaginary displacement front illustrated by thresholding time-of-flight iniside the sweep regions for the base case.

regions of low flow (light gray). In the figure, we see that wells I1 and P2 are connected by a high-flow region, which explains the early breakthrough we observed in Figure 13.26. The existence of high-flow regions can also be seen from the F–Φ diagram and the Lorenz coefficient of 0.273.

The interactive diagnostic tool has functionality that lets you modify the well controls and if needed, add new wells or remove existing ones. We now use this functionality to try to manually improve the volumetric sweep of the reservoir, much in the same way as we did for a layer of SPE 10 in Section 13.4.2. We start by reducing the high flow rate in the region influenced by I1 and P2. That is, we increase the pressure in P2 to, say, 130 bar to decrease the inter-well pressure drop. The resulting setup, referred to as "Case 1," gives more equilibrated flow paths, as you can see from the Lorenz coefficient and the upper-right plot in Figure 13.27.

Case 2 further decreases the pressure in P3 to 80 bar to increase the flow in the I2–P3 region. As a result of these two adjustments to the well pressures, we have reduced the stagnant region north of P2 and also diminished the clear flow-divide that extended from the south of the reservoir to the region between P2 and P3. To also sweep the large unswept region east of P3, we can use infill drilling to introduce a new well to the southeast of this region, just north of the sealing fault. Since the new well is quite close to the

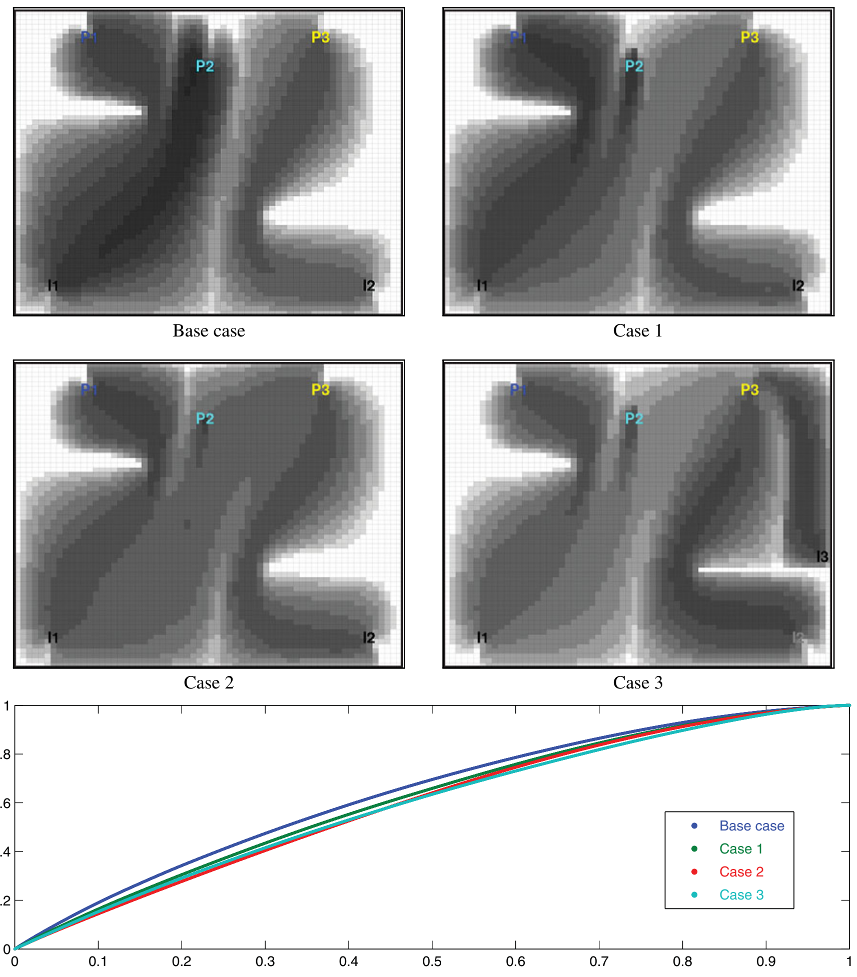

Figure 13.27 In Cases 1 and 2, well controls have been manually adjusted from that of the base case (see Table 13.2) to equilibrate total travel time throughout the reservoir. Case 3 includes infill drilling of an additional injector. The bottom plot shows the corresponding F–Φ curves.

existing producer, we should assign a relatively low pressure to avoid introducing too high flow rates.

In Case 3, we let the well operate at 140 bar and at the same time we have increased the pressure in I2 to 220 bar. Figure 13.28 shows snapshots of the advancing front at the same instances in time as was used in Figure 13.26 for the base case. Altogether, the well configuration of Case 3 gives significantly improved areal sweep and reduces the Lorenz coefficient to 0.189. We can therefore expect this configuration to give better displacement if the setups were rerun with a multiphase simulator.

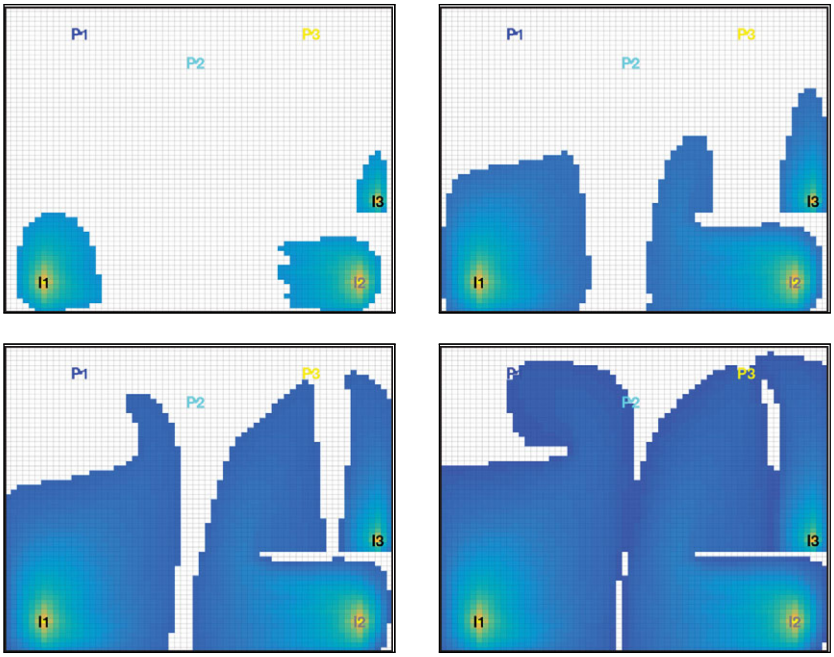

Figure 13.28 Evolution of an imaginary displacement front illustrated by thresholding time-of-flight iniside the sweep regions for Case 3.

You can find a more detailed description of how you should use the interactive GUI to perform the experiments just described in the `interactiveSimple.m` script of the `book` module. I encourage you to use the script to familiarize yourself with interactive flow diagnostics. Are you able to make further improvements?

13.5.2 SAIGUP: Flow Patterns and Volumetric Connections

The previous example was highly simplified and chosen mainly to illustrate the possibilities that lie in the interactive use of flow diagnostics. In this example, we take a closer look at the volumetric connection in the SAIGUP example from Section 5.4.4 on page 169. We start by rerunning the example script to set up the simulation model and compute a flow field:

```
saigupWithWells; close all
clearvars -except G rock W state
```

Henceforth, we only need the geological model, description of the wells, and the reservoir state. We clear all other variables and close all plots produced by the script. We can pass the reservoir state to the interactive GUI and use it in pure postprocessing mode:

```
interactiveDiagnostics(G, rock, W, 'state', state, 'computeFlux', false);
```

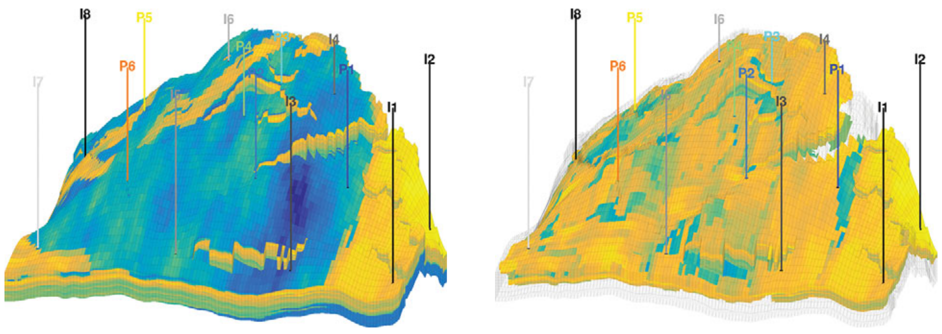

Figure 13.29 Horizontal permeability ($\log_{10} K_x$) for the SAIGUP model. The left plot shows the full permeability field, while the right plot only shows the permeability in cells that have a total residence time less than 100 years. (The reservoir is plotted so that the north–south axis goes from left to right in the figure.)

In this mode, we are not able to edit any well definitions and recompute fluxes.

In Figure 5.11 on page 173, we saw that, even though injectors and producers are completed in all 20 grid layers of the model, there is almost no flow in the bottom half of the reservoir. A more careful inspection shows that there is almost no flow in the upper layers in most of the reservoir either because the best sand quality is found in the upper-middle layers of the reservoir. Figure 13.29 confirms this by comparing the permeability in the full model with the permeability in cells having a residence time less than 100 years.

Because each injector is controlled by a total fluid rate, large fluid volumes are injected in completions connected to good quality sand, while almost no fluid is injected into zones with low permeability and porosity. You can see this in Figure 13.30, which shows overall and cumulative wellallocation factors for four of the injectors. (These plots are conceptually similar to what you would obtain by running a production logging tool for all these wells.) Injectors I3 and I4 are completed in the southern part of the reservoir, and here low-quality sand in the bottom half of the reservoir leads to almost negligible injection rates in completions 11–20. In a real depletion plan, the injectors would probably not have been completed in the lower part of the sand column. Injector I5 located to the west in the reservoir is completed in a column with low permeability in the top four and the bottom layer, high permeability in Layers 6–9, and intermediate permeability in the remaining layers. Hence, almost no fluid is injected through the top four completions, which hence are redundant. Finally, injector I6 is completed in a column with poor sand quality in the top three layers, high permeability in Layers 4–9, and intermediate permeability in the remaining layers.

Figure 13.30 also shows the flux allocation for all well pairs in the reservoir. Each curved line corresponds to a injector–producer connection, the line color signifies the producer, and the percentage signifies the fraction of the total flux from each injector that goes to the different producers. We have truncated the connections to only pairs that correspond

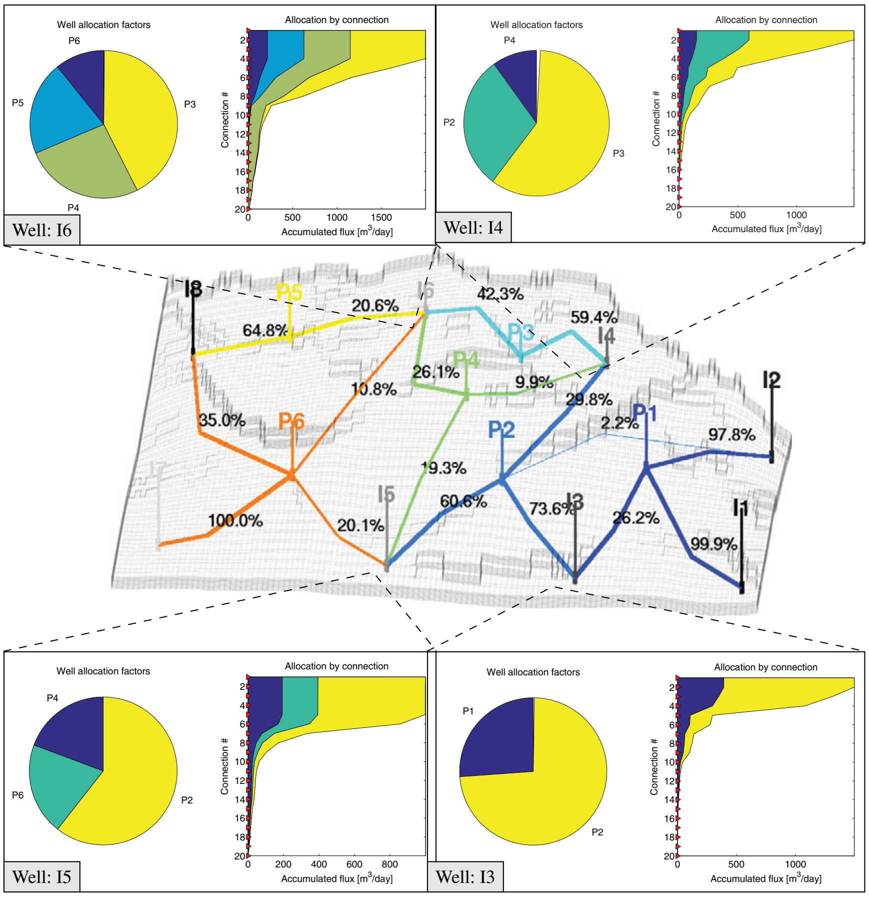

Figure 13.30 Flux allocation for all well pairs of the SAIGUP model and cumulative well-allocation factors from bottom to top layer for injectors I3, I4, I5, and I6.

to at least 1% of the flux. This explains why not all fractions sum up to unity. Let us take injector I4 as an example. Figure 13.31 shows two plots of the influence region for this injector. From the well allocation plots, we have already seen that I4 is connected to producers P2 to P4. Almost all the completions of these wells lie inside the region colored by I4. Producer P1, on the other hand, is only completed in a single cell inside the colored region, and this cell is in the top layer of the reservoir where the sand quality is very poor. It is therefore not clear whether P1 is actually connected to I4 or if this weak connection is a result of inaccuracies in the computation of color concentrations. Since we only use a first-order discretization, the color concentrations can suffer from a significant amount of numerical smearing near flow divides, which here is signified by blue-green colors.

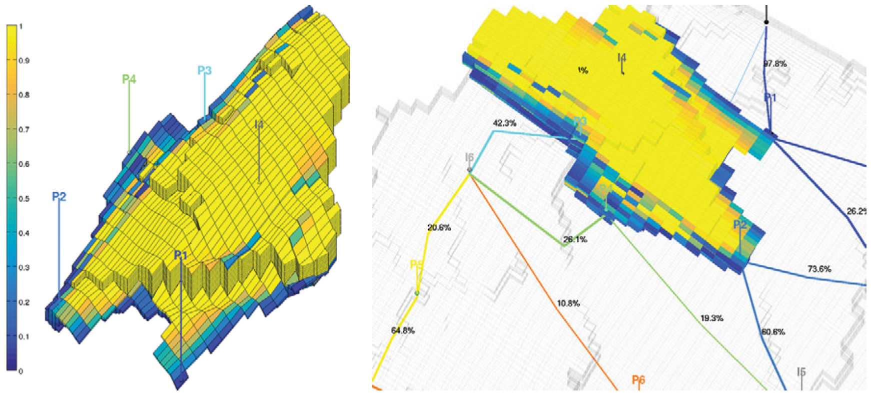

Figure 13.31 Plot of the influence region for well I4.

Computer exercises

13.5.1 Change all the injectors to operate at a fixed bottom-hole pressure of 300 bar. Does this significantly change the flow pattern in the reservoir? Which configuration do you think is best?

13.5.2 Use the flow diagnostic tool to determine well completions in the SAIGUP model that have insignificant flow rate, eliminate these well completions, and rerun the model. Are there any apparent changes in the flux allocation and volumetric connections?

13.5.3 Consider the model given in `makeAnticlineModel.m`. Can you use flow diagnostics to suggest a better well configuration?

13.5.4 Go back to the SPE 9 model from Section 12.4.4. Would it make sense to use flow diagnostics to analyze the flow patterns of this model? If so, how would you do it?