Chapter 1

Two-phase flow

1.1 Two-phase flow.

When simulating two phases within a porous medium, we often consider them as separated phases, i.e., they are inmiscible and there is no mass transfer between the phases. The contact between phases is known as the interface between two reservoir fluids phases.

While modeling two phases, we usually consider one of the fluids as the wetting phase (w) that wets the porous medium more than the other. The other phase is consider as non-wetting phase (n). In the case of a water-oil system, water is often the wetting phase.

For a reservoir, the density of each phase varies, the gas is less dense than the oil and water. Therefore, the gas is found on top of the reservoir. For the oil-water system, water's density is larger than oil's, then water is found on the bottom of the reservoir.

The saturation of a phase S_α, is the fraction of void space filled with that phase in a porous medium. If there are two phases present in the porous medium, these fluids fill completely the empty space, which is expressed by the following relation.

$$S_n + S_w = 1 \tag{1.1}$$

The surface tension and the curvature of the interface between the fluids causes a difference in pressure between the two phases. The pressure in the wetting fluid is less than in the nonwetting fluid. This difference in pressures is known as the capillary pressure, p_c, and it's function of saturation:

$$p_c(S_w) = p_n - p_w. \tag{1.2}$$

As mentioned before, the pressure in the non-wetting fluid is higher than the pressure in the wetting fluid, therefore the capillary pressure is always positive. The relation between the capillary pressure and saturation is obtained as an empirical model based on experiments. However, each core sample will have different capillary curve due to the difference in pore-size distributions, porosity and permeability. To normalize the measured data, it's common to use a so-called Leverett J-function, which takes the following form:

$$J(S_w) = \frac{P_c}{\sigma cos\theta} \sqrt{\frac{K}{\phi}}, \qquad (1.3)$$

where σ is the surface tension and θ the contact angle measured in the laboratory for an specific rock and fluid system.

A model that relates the capillary pressure and water saturation in a partially saturated media (water and air media) is:

$$\hat{S}_w = \begin{cases} (p_c/p_e)^{-n_b} & \text{if } p_c > p_e \\ 1 & p_c \le p_e \end{cases}$$

where p_e is the entry pressure of air, and n_b is related to the pore-size distribution. This model was proposed by Brooks and Corey. Another model was proposed by Genuchten:

$$\hat{S}_w = \left(1 + (\beta_g p_c)_g^n\right)^{-m_g}, \qquad (1.4)$$

where β_g is related to the average size of pores and n_g and m_g are related to the pore size distribution.

Relative permeability

When more than one phase is present in the pore space, each phase α will experience an effective permeability K_α^e that is less than the absolute permeability K. Due to interfacial tensions, the sum of all the phase permeabilities is less than one.

$$\sum_\alpha K_\alpha^e < K.$$

The relative permeability for an isotropic medium is defined as:

$$k_{r\alpha} = K_\alpha^e/K.$$

The relative permeabilities will generally be functions of saturation, generally nonlinear. It is common to use analytic relationships to represent relative permeabilities. These are usually stated using normalized or effective saturations $\hat{S}_w$. The simplest model possible is called the Corey model:

$$k_{rw} = (\hat{S}_w)^{n_w} k_w^0,$$
$$k_{ro} = (1 - \hat{S}_w)^{n_n} k_o^0. \qquad (1.5)$$

where $n_w > 1$, $n_o > 1$ and k_α^0 are fitting parameters.

The Brooks-Corey functions are also used.

$$k_{rw} = (\hat{S}_w)^{n_1 + n_2 n_3},$$
$$k_{ro} = (1 - \hat{S}_w)^{n_1} [1 - (\hat{S}_w)^{n_2}]^{n_3}.$$

where, $n_1 = 2$, $n_2 = 1 + 2/n_b$ and $n4 = 1$ for the Brooks–Corey–Burdine model and $n_1 = \eta$, $n_2 = 1 + 1/n_b$ and $n3 = 2$ for the Brooks–Corey–Mualem model. For the van Genuchten capillary functions $m_g = 1 - 1/n_g$) we have:

$$k_{rw} = \hat{S}_w^2 [1 - (1 - \hat{S}_w^{1/m_g})^{m_g}],$$
$$k_{ro} = (1 - \hat{S}_w)^2 [1 - (\hat{S}_w^{1/m_g})]^{m_g},$$

which is called the Genuchten-Burdine model.

As in the single-phase case, the governing equations for two-phase flow in a porous medium are the mass conservation and Darcy's law. The mass balance equations for each phase α are given by:

$$\frac{\partial(\phi\rho_\alpha S_\alpha)}{\partial t} + \nabla \cdot (\rho_\alpha \mathbf{v}_\alpha) = \rho_\alpha q_\alpha,$$

Darcy's law is:

$$\mathbf{v}_\alpha = -\frac{k_{r\alpha}}{\mu_\alpha}K(\nabla p_\alpha - \rho_\alpha g \nabla z).$$

To simplify notation, the phase mobilities ($\lambda_\alpha = Kk_{r\alpha}/\mu_\alpha$) or relative phase mobilities ($\lambda_\alpha = \lambda_\alpha K$) are used. Using the discrete derivative operators and backward discretization of temporal derivatives, the resulting system of discrete equations is:

$$\frac{(\phi\rho_\alpha \mathbf{S}_\alpha)^{n+1} - (\phi\rho_\alpha \mathbf{S}_\alpha)^n}{\Delta t^n} + div(\rho_\alpha \mathbf{v}_\alpha)^{n+1} = \mathbf{q}_\alpha^{n+1},$$

$$\mathbf{v}_\alpha^{n+1} = -\frac{\mathbf{k}_{r\alpha}}{\mu_\alpha}\mathbf{K}[grad(\mathbf{p}_\alpha^{n+1}) - g\rho_\alpha^{n+1}grad(\mathbf{z})].$$

Immiscible two-phase flow

For the case of immiscible fluids, the flow equations are:

$$\frac{\partial(\phi\rho_w S_w)}{\partial t} + \nabla \cdot (\rho_w \mathbf{v}_w) = \rho_w q_w, \tag{1.6}$$

$$\frac{\partial(\phi\rho_n S_.)}{\partial t} + \nabla \cdot (\rho_n \mathbf{v}_n) = \rho_n q_n. \tag{1.7}$$

Pressure formulation

We can choose the pressures as the primary unknowns, then the saturations are expressed as functions of pressure. Assuming the capillary pressure has a unique inverse function $\hat{S}_w = P_c^{-1}(p_c)$, then we have:

$$S_w = \hat{S}_w(p_n - p_w) \qquad S_n = 1 - \hat{S}_w(p_n - p_w),$$

then equations (1.6) and (1.7) can be written as:

$$\frac{\partial(\phi\rho_w\hat{S}_w)}{\partial t} + \nabla \cdot (\rho_w\frac{Kk_{rw}(\hat{S}_w)}{\mu_w}(\nabla p_w - \rho_w g \nabla z)) = \rho_w q_w, \tag{1.8}$$

$$\frac{\partial(\phi\rho_n(1-\hat{S}_w))}{\partial t} + \nabla \cdot (\rho_n\frac{Kk_{rn}(\hat{S}_w)}{\mu_n}(\nabla p_n - \rho_n g \nabla z)) = \rho_n q_n. \tag{1.9}$$

Previous system is highly coupled and strongly nonlinear.

Fractional flow formulation

Part of the non linearity of previous formulation can be eliminated if the system is expressed in terms of one phase pressure and one phase saturation. A common choice is to use p_n and S_w which gives the following system

$$\frac{\partial(\phi\rho_w S_w)}{\partial t} + \nabla \cdot (\rho_w\frac{Kk_{rw}}{\mu_w}(\nabla p_n - \nabla P_c(S_w) - \rho_w g \nabla z)) = \rho_w q_w, \tag{1.10}$$

$$\frac{\partial(\phi\rho_n(1-S_w))}{\partial t} + \nabla \cdot (\rho_n \frac{Kk_{rn}}{\mu_n}(\nabla p_n - \rho_n g\nabla z)) = \rho_n q_n. \qquad (1.11)$$

Incompressible flow

For incompressible flow, only the Saturation S is a function of time and the fluid densities ρ_α are constant. Therefore, the mass-balance equations are:

$$\phi\frac{\partial(S_\alpha)}{\partial t} + \nabla \cdot (\mathbf{v}_\alpha) = q_\alpha. \qquad (1.12)$$

The total Darcy velocity is defined as the sum of the velocity in the wetting and non wetting phases, it can be expressed in terms of the non-wetting phase as:

$$\mathbf{v} = \mathbf{v}_w + \mathbf{v}_n = -\lambda_n\nabla p_n - \lambda_w\nabla p_w + (\lambda_n\rho_n + \lambda_w\rho_w)g\nabla z$$
$$= -(\lambda_n + \lambda_w)\nabla p_n + \lambda_w\nabla p_c + (\lambda_n\rho_n + \lambda_w\rho_w)g\nabla z$$

if we add the two continuity equations and use the relationship $S_n + S_w = 1$ we have:

$$\phi\frac{\partial(S_w + S_n)}{\partial t} + \nabla \cdot (\mathbf{v}_w + \mathbf{v}_n) = q_n + q_w.$$

Using $\lambda = \lambda_n + \lambda_w = \lambda K$ as the total mobility and $q = q_n + q_w$ as the total source, for the total Darcy velocity we have:

$$-\nabla \cdot (\lambda K\nabla p_n) = q - \nabla[\lambda_w\nabla p_c + (\lambda_n\rho_n + \lambda_w\rho_w)g\nabla z].$$

Multiplying each phase velocity by the relative mobility of the other phase and subtracting the result we obtain:

$$\lambda_n\mathbf{v}_w - \lambda_w\mathbf{v}_n = \lambda\mathbf{v}_w - \lambda_w\mathbf{v}$$
$$= \lambda_w\lambda_n K[\nabla p_c + (\rho_w - \rho_n)g\nabla z].$$

Therefore, for $\mathbf{v}_w$ we have

$$\mathbf{v}_w = \frac{\lambda_w}{\lambda}\mathbf{v} + \frac{\lambda_w\lambda_n}{\lambda}K[\nabla p_c + (\rho_w - \rho_n)g\nabla z].$$

Using the velocity computed above, for the wetting phase we have:

$$\phi\frac{\partial(S_w)}{\partial t} + \nabla \cdot [f_w(\mathbf{v} + \lambda_n\Delta\rho g\nabla z)] = q_w - \nabla \cdot (f_w\lambda_n p_c\nabla S_w). \qquad (1.13)$$

with $\Delta\rho = \rho_w - \rho_n$ and the fractional flow function f_w:

$$f_w = \frac{\lambda_w}{\lambda_n + \lambda_w},$$

is the fraction of the total flow that consists of the wetting fluid, and

$$\mathbf{v} = -\lambda(\nabla p_n - f_w\nabla p_w - (f_n\rho_n + f_w\rho_w)g\nabla z)$$

The coupling between the elliptic pressure equation and the parabolic saturation equation is much weaker than the coupling between the two continuity equations in the two-pressure formulation. In the pressure equation, the coupling to saturation appears explicitly in the effective mobility that makes up the

4

variable coefficient in the Poisson problem and on the right-hand side through the phase mobilities and the derivative of the capillary function. In Equation (1.13), the saturation is only indirectly coupled to the pressure through the total Darcy velocity.

The system of PDEs can then be reformulated so that it consists of an elliptic equation for fluid pressure and one or more transport equations. These transport equations are generally parabolic, but have a strong hyperbolic character. Since the pressure and saturations equations have very different mathematical characteristics, it is natural to solve them in consecutive substeps. Sequential solution procedures and in- compressible flow models are popular in academia and for research purposes, but are less used in industry. To the extent simulations are used for practical reservoir engineering, they are mainly based on compressible equations and solution procedures in which flow and transport are solved as a coupled system. Such approaches are very robust and particularly useful for problems with large variations in time constants or strong coupling between different types of flow mechanisms.

Sequential solution procedures

The two-phase, incompressible model will be solved using the fractional-flow formulation. This fractional flow model consists of an elliptic pressure equation

$$\nabla \cdot \mathbf{v} = q, \qquad \mathbf{v} = -\lambda(\nabla p_n - f_w \nabla p_c - (f_n \rho_n + f_w \rho_w)g\nabla z) \qquad (1.14)$$

and a parabolic transport equation (1.13)

$$\phi \frac{\partial(S_w)}{\partial t} + \nabla \cdot [f_w(\mathbf{v} + \lambda_n(\Delta \rho g \nabla z) + \nabla P_c)] = q_w. \qquad (1.15)$$

Where the capillary pressure $p_c = p_w - p_n$ is assumed to be a known function P_c of the wetting saturation S_w. The transport equation becomes hyperbolic whenever P_c is zero.

In the standard sequential solution procedure, the system above, is evolved in time using a set of discrete time steps Δt_i . We assume that $p, \mathbf{v}$, and S_w are all known at time t and that we want to evolve the solution to time $t + \Delta t$.

At the beginning of the time step, we first assume that the saturation S_w is fixed. This means that the parameters λ, f_w, and f_n are functions of the spatial variable $\mathbf{x}$, and hence we can use this equation only to update the pressure p_n and the Darcy velocity $\mathbf{v}$. Then, $\mathbf{v}$ and p_n are held fixed while equation 1.15 is evolved a time step Δt to define an updated saturation $S_w(\mathbf{x}, t + \Delta t)$. This saturation is then held fixed when we update p_n and $\mathbf{v}$ in the next time step, and so on. Some authors refer to this solution procedure as an operator splitting method since the solution procedure effectively splits the overall solution operator of the flow model into two parts that are evolved in consecutive substeps. Likewise, some authors refer to the sequential solution procedure as IMPES, which is short-hand for implicit pressure, explicit saturation. Using the name IMPES is strictly speaking only correct if the saturation evolution is approximated by a single time step of an explicit transport solver.

Pressure solvers

The pressure Equation (1.14) for incompressible, multiphase flow is time dependent. This time dependence comes as the result of three factors:

1. K/μ being replaced by the total mobility $\lambda(S_w)$, which depends on time through the saturation $S_w(\mathbf{x}, t + \Delta t)$,

2. the constant density ρ being replaced by a saturation-dependent quantity $\rho_w f_w(S_w) + \rho_n f_n(S_w)$, and

3. the source term q being replaced by a saturation-dependent source term $q - \nabla \lambda_w(S_w) \nabla P_c(S_w)$.

However, once S_w is held fixed in time, all three quantities become functions of $\mathbf{x}$ only, and we hence end up again with an elliptic Poisson-type equation having the same spatial variation.

Saturation solvers

The saturation equation depends on the time, using backward Euler discretization for the time derivative in Equation 1.15, we have:

$$\phi \frac{(S_w^{n+1} - S_w^n)}{\Delta t} + \nabla \cdot [f_w(S_w)(\mathbf{v} + \lambda_n(S_w)(\Delta \rho g \nabla z + \nabla P_c(S_w)))] = q_w, \quad (1.16)$$

or

$$S_w^{n+1} = S_w^n - \frac{\Delta t}{\phi} \nabla \cdot [f_w(S_w)(\mathbf{v} + \lambda_n(S_w)(\Delta \rho g \nabla z + \nabla P_c(S_w)))] + q_w,$$

which can be computed explicitly:

$$S_w^{n+1} = S_w^n - \mathcal{F}(S_w^n, S_w^n),$$

or implicitly:

$$S_w^{n+1} = S_w^n - \mathcal{F}(S_w^{n+1}, S_w^n).$$

If we use the implicit scheme, the system is nonlinear and depends on the saturation at time step n and $n+1$.

The discretization is implemented with the following residual form for each cell Ω_i,

$$F_i(s, r) = s_i - r_i + \frac{\Delta t}{\phi_i |\Omega_i|} [H_i(s) - max(q_i, 0) - min(q_i, 0) f(S_i)], \quad (1.17)$$

where s and r are cell-averaged quantities and subscript i refers to the cell the average is evaluated in. The sum of the interface fluxes for cell i

$$H_i(s) = \sum_k \frac{\lambda_w^u(s_i, s_k)}{\lambda_w^u(si, sk) + \lambda_n^u(si, sk)} [v_{ik} + \lambda_n^u(s_i, s_k)(g_{ik} + P_{ik})], \quad (1.18)$$

is computed using the single-point upstream mobility-weighting, whereas the fractional flow function f in the source term is evaluated from the cell average of S in cell Ω_i. The explicit scheme is given as $S^{n+1} = S^n - F(S^n, S^n)$ and the implicit solution is obtained solving $F(S^{n+1}, S^n) = 0$. The residual equations (1.17) for all cells in vector form

$$\mathbf{F}(\mathbf{s}) = \mathbf{s} - \mathbf{S} + \frac{\Delta t}{\phi_i |\Omega|} [\mathbf{H}(\mathbf{s}) - \mathbf{Q}^+ - \mathbf{Q}^- \mathbf{f}(\mathbf{S})] = \mathbf{0}. \quad (1.19)$$

Here, s is the unknown state at time tf and S is the known state at the start of the time step. The nonlinear system can be solved with NR method, where, for the $(k + 1)$-th iteration we have:

6

$$\mathbf{J}(\mathbf{S}^k)\delta\mathbf{S}^{k+1} = -\mathbf{F}(\mathbf{S}^k; \mathbf{S}^n), \qquad \mathbf{S}^{k+1} = \mathbf{S}^k + \delta\mathbf{S}^{k+1},$$

where $\mathbf{J}(\mathbf{S}^k) = \frac{\partial\mathbf{F}(\mathbf{S}^k;\mathbf{S}^n)}{\partial\mathbf{S}^k}$ is the Jacobian matrix, and $\delta\mathbf{S}^{k+1}$ is the NR update at iteration step $k+1$.

Therefore, the linear system to solve is:

$$\mathbf{J}(\mathbf{p}\mathbf{S}^k)\delta\mathbf{S}^{k+1} = \mathbf{b}(\mathbf{S}^k). \tag{1.20}$$

with $\mathbf{b}(\mathbf{S}^k)$ being the function evaluated at iteration step k, $\mathbf{b}(\mathbf{S}^k) = -\mathbf{F}(\mathbf{S}^k; \mathbf{S}^n)$.
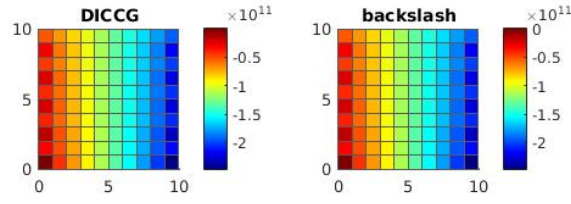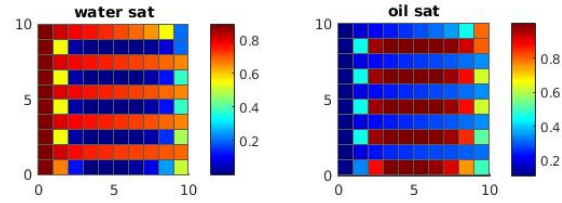
1.2 Experiments

Figure 1.1: Solution, pressure
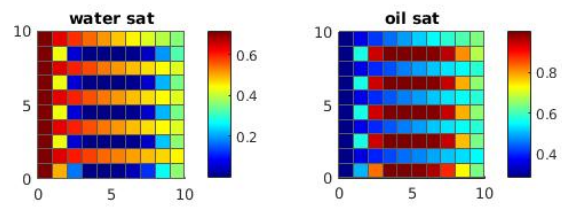

Figure 1.2: solution, saturation explicit

Figure 1.3: Solution,saturation implicit

Appendix 1. MRST

Two-phase flow

When going from a single-phase to a multiphase flow model, the most prominent changes take place in the fluid model. It is this model that generally will tell your solver how many phases are present and how these phases affect each other when flowing together in the same porous medium. To describe an incompressible flow model, we need to know the viscosity and the constant density of each fluid phase, as well as the relative permeabilities of the fluid phases. If the fluid model includes capillary forces, we also need one or more functions that specify the capillary pressure as function of saturation. The most basic multiphase fluid model in MRST is the following,

$$
\begin{aligned}
fluid \quad &= initSimpleFluid(\ \text{'mu'}, [1, 10] * centi * poise, ... \\
&\text{' rho ', } [1014, 859] * \text{kilogram /meter \^{}3}, ... \\
&\text{' n ' , } [2, 2]);
\end{aligned}
$$

which implements a simplified version of the Corey model (see Equation 1.5) in which the residual saturations S_{wr} and S_{nr} are assumed to be zero and the end-points and scaled to unity, so that $k_{rw} = (S_w)^{n_w}$ and $S_{rn} = (1 - S_w)^{n_n}$

```
mu = fluid.properties();% gives mu_w and mu_n
[mu, rho] = fluid.properties(); % .... plus rho_w and rho_n
```

New to multiphase flow is the relperm function, which takes a single fluid saturation or an array of fluid saturations as input and outputs the corresponding values of the relative permeabilities. Plotting the relative permeability curves of the fluid object can be achieved by the following code

```
s=linspace(0,1,20)';
kr = fluid.relperm(s);
plot(s,kr (:,1), ' −s' ,s,kr (:,2), ' −o');
```

The relperm function can also return the first and second derivatives of the relative permeability curves when called with two or three output arguments. The basic fluid model does not have any capillary pressure. To also include this effect, one can use another fluid model, which adds a capillary function that

```
fluid = initSimpleFluidPc('pc scale', 2*barsa);
```

assumes a linear relationship $P_c(S) = C(1 - S)$.

The capillary pressure function pc is evaluated using a state object and not a saturation. This simple function can be extended to include Leverett J- function scaling by using the fluid object generated by $initSimpleFluidJfunc$. The incomp module also implements the general Corey model with end- point scaling k_α^0 and nonzero residual saturations S_{wr} and S_{nr}.

Explicit solver

The incomp module offers the following explicit transport solver

$$state = explicitTransport(state, G, tf, rock, fluid,' mech1', obj1, ...);$$

which evolves the saturation given in the state object a step tf forward in time. The function requires a complete and compatible model description consisting of a grid structure G, petrophysical properties rock, and a fluid model fluid. The state object must contain the correct number of saturations per cell and an incompressible flux field that is consistent with the global drive mechanisms given by the mech argument ('src', 'bc', and/or 'wells') using correctly specified objects obj. The input value of state must be the output value of a previous call to an incompressible solver like incompTPFA, incompMPFA, or incompMimetic. In addition, the function takes a number of optional parameters that determine whether the time steps are prescribed by the user or to be automatically computed by the solver. The solver can also ignore the Darcy flux and work as a pure gravity segregation solver if the optional parameter onlygrav is set to true. Finally, the solver will issue a warning if the updated saturation value is more than satwarn outside the interval [0, 1] of physically meaningful values (default value: sqrt(eps)).

The implicit and the explicit solvers involve many of the same operations and formulas used for the spatial discretizations. To avoid duplication of code we have therefore introduced a private help function that implements the residual

$$[F, Jac] = twophaseJacobian(G, state, rock, fluid,' pn1', pv1, ...);$$

form (10.3) and its Jacobian matrix J = dF and returns these as two function handles F and Jac. Using this function, the key lines of the explicit saturation solver read,

```
F = twophaseJacobian(G, state, rock, fluid,' wells', opt.wells, ...);
s = state.s(:, 1);
t = 0;
while t ¡ tf,
    dt = min(tf − t, getdt(state));
    s(:) = s − F(state, state, dt);
    t = t + dt;
     s = correct_saturations(s, opt.satwarn);
    state.s = [s, 1 − s];
end
```

The function $correct_saturations$ ensures that the computed saturations stay inside the interval of physically valid states.

Implicit solver

The implicit solver has the same user interface and parameter requirement as the explicit solver In addition, there are optional parameters that control the

$$state = explicitTransport(state, G, tf, rock, fluid,' mech1', obj1, ...);$$

Newton–Raphson method used to solve for S^{n+1}.