Richards Examples

Andy Wilkins
CSIRO

January 13, 2015

Contents

1 Introduction

The Richards' equation describes slow fluid flow through a porous medium. This document outlines input-file examples for the Richards MOOSE code, drawing upon the test suite, and provides guidelines for creating models that run smoothly. There are two other accompanying documents: (1) The theoretical and numerical foundations of the code, which also describes the notation used throughout this document; (2) The test suite, which describes the benchmark tests used to validate the code.

Each example is located in the *test* directory, which has path

```
<install_dir>/moose/modules/richards/tests
```

2 User Object examples

In the test suite there are a number of tests that allow easy exploration of the functional form of the User Objects. These are:

1. `relperm.i` outputs the relative permeability as a function of effective saturation. The user may specify which particular form of relative permeability to output, and the appropriate parameters of interest.

2. `density.i` outputs the fluid density as a function of fluid pressure. The user may specify which particular form of density to output, and the appropriate parameters of interest.

3. `seff1.i` outputs the effective saturation for a single-phase situation as a function of porepressure (if porepressure is positive then the effective saturation is unity). The user may specify which particular form of effective saturation to output, and the appropriate parameters of interest.

4. `seff2.i` outputs the effective saturation for a 2-phase situation as a function of differences in porepressure (the difference is, say, $P_{\rm gas} - P_{\rm water}$, which is positive). The user may specify which particular form of effective saturation to output, and the appropriate parameters of interest.

In all cases, an exodus file is output that can be used to generate plots of the user objects. Also a CSV file is output that contains the user object evaluated at a specific point of interest.

3 Convergence and convergence criteria

As a general rule, the formulation of multiphase flow implemented in MOOSE is quite suitable to solve many different problems. However, there are some situations where the implementation is not optimal, such as the tracking of fronts. MOOSE may converge in these instances, but may take unacceptably short time steps, or it may not even converge at all. In these cases it is probably best to either modify the problem to something more suitable, or use another code.

For example, in simulations with sharp fronts, the user should ask whether it is truly necessary to accurately track the sharp front, or whether similar results could be taken by smoothing the fronts by using slightly different initial conditions, and/or by modifying the effective-saturation relationships, by making the van Genuchten parameter α smaller, for instance. If tracking sharp fronts is vital to the problem then a code specifically designed to solve such problems will do it better than MOOSE, or perhaps MOOSE could be used with an ALE front-tracking algorithm.

Below I list some general pointers that may help with problems that MOOSE is struggling with.

- External fluxes that turn off or on too quickly are bad. Simialrly, those that have large discontinuities in their derivatives can cause convergence problems. Try to define "smooth" versions of these as inputs. Discontinuities like these often manifest themselves in non-convergence of the nonlinear iterations.

- Effective saturation curves that are too "flat" are not good. For example, the van Genuchten parameter $m = 0.6$ almost always gives better convergence than $m = 0.9$.

- Effective saturation curves that are too "low" are not good. For example, the van Genuchten parameter $\alpha = 10^{-6}\,\mathrm{Pa}^{-1}$ almost always gives better convergence than $\alpha = 10^{-3}\,\mathrm{Pa}^{-1}$. Compare, for instance, the tests `buckley_leverett/bl21.i` and `buckley_leverett/bl22.i`.

- Any discontinuities in the effective saturation, or its derivative, are bad. I suggest using van Genuchten parameter $\alpha \geq 0.5$ for problems with both saturated and unsaturated zones if the van Genuchten relative permeability relation is used.

- Highly nonlinear relative permeability curves make convergence difficult in some cases. For instance, a "power" relative permeability curve with $n = 20$ is much worse numerically than with $n = 2$. See if you can reduce the nonlinearity in your curve.

- Any discontinuities in the relative permeability, or its derivative, are bad. For most curves coded into MOOSE this is not an issue, but I recommend the `RichardsRelPermVG1` curve over `RichardsRelPermVG`, since the former is smooth around $S_{\mathrm{eff}} = 1$. See `recharge_discharge/rd01.i` in the tests directory for an example of this. I also suggest using the "power" covers over the "VG" curves.

- In multiphase problems if one phase completely disappears, MOOSE may not converge, as discussed more fully in Chapter 6. To avoid this:
  1. The fully-upwind kernel and boundary fluxes and dirac sources can be used. If the immobile saturation of the phase is nonzero, then it probably won't disappear, as the fully-upwind approach will not, in theory, allow fluid to exit from a node if the relative permeability is zero. However, numerical imprecision can lead to phase disappearance.
  2. A nonzero residual saturation can be used. This means that for $dt \to 0$ the Jacobian matrix will be nonsingular. (If the residual saturations are zero then the Jacobian is singular for $dt \to 0$.) Then in most cases the problematic node will fill with a little amount of the phase in the next time step.
  3. A "shifted" van Genuchten capillary suction curve may be used in difficult multiphase problems.

- When porosity is a function of time, small time steps may be necessary, especially in multiphase situations. MOOSE will converge, but may need quite a few nonlinear iterations (maybe about 20), and the PETSc tolerances should be set quite tight, otherwise MOOSE might "converge" to a crazy solution that causes difficulties in subsequent time steps.

- Check whether the linear or nonlinear solvers are causing the problem. If it is the latter, and you have addressed the potential problems above (most particularly discontinuities, and phase disappearence), then probably your problem is just highly nonlinear. However, you can often address problems with the *linear* solver not converging quickly by choosing different preconditioners and ksp methods. I typically use one of:
  1. `-ksp_type=gmres -pc_type=bjacobi`
  2. `-ksp_type=gmres -pc_type=asm -sub_pc_type=lu -sub_pc_factor_shift_type=NONZERO`

Choosing reasonable convergence criteria is very important. The Theory Manual contains a section that explains the *minimum* residual that a user can expect to obtain in a model. However, this minimum is usually much smaller than what is important from a practical point of view. If a tiny residual is chosen, MOOSE will spend most of its time changing pressure values by tiny amounts as it attempts to converge to the tiny residual, and this is a waste of compute time. This problem may be amplified if adaptive time-stepping is used since MOOSE doesn't realise that most of the compute time is spent "doing nothing", so keeps the timestep very small. So, here are some guidelines for choosing an appropriate tolerance on the residual.

1. Determine an appopriate tolerance on what you mean by "steadystate". For instance, in a single-phase simulation with reasonably large constant fluid bulk modulus, and gravity acting in the $-z$ direction, the steadystate solution is $P = -\rho_0 g z$ (up to a constant). In the case of water, this reads $P = -10000z$. Instead of this, suppose you would be happy to say the model is at steadystate if $P = -(\rho_0 g + \varepsilon)z$. For instance, for water, $\varepsilon = 1\,\mathrm{Pa.m^{-1}}$ might be suitable in your problem. Then recall that the residual is just

$$R = \left| \int \nabla_i \left(\frac{\kappa_{ij}\kappa_{\mathrm{rel}}\rho}{\mu} (\nabla_j P + \rho g_j) \right) \right| \tag{3.1}$$

Evaluate this for your "steadystate" solution. For instance, in the case of water just quoted, $R = V|\kappa|\rho_0/\mu\varepsilon = V|\kappa| \times 10^6\varepsilon$, where: V is the volume of the finite-element mesh, and I have inserted standard values for ρ_0 and μ.

2. In the previous step, an appropriate tolernace on the residual was given as $V|\kappa|\rho_0\varepsilon/\mu$. However, this is often too large because of the factor of V. The previous step assumed that the solution was incorrect by a factor, ε, which is constant over the entire mesh. More commonly, there is a small region of the mesh where most of the interesting dynamics occurs, and the remainder of the mesh exists just to provide reasonable boundary conditions for this "interesting" region. The residual in the "boring" region can be thought of as virtually zero, while the residual in the "interesting" region is $V_{\text{interesting}}|\kappa|\rho_0\varepsilon/\mu$. This is smaller than the residual in the previous step, so provides a tighter tolerance for MOOSE to strive towards.

3. In the previous steps, I've implicitly assumed κ is constant, ρ is virtually constant at ρ_0, only a single-phase situation, etc. In many cases these assumptions are not valid, so the integral of Eqn (3.1) cannot be done as trivially as in the previous steps. In these cases, I simply suggest to build a model with initial conditions like $P = -(\rho_0 g + \varepsilon)z$, and just see what the initial residual is. That will give you an idea of how big a reasonable residual tolerance should be.

4 Two-phase, almost saturated

If a two-phase model has regions that are fully saturated with the "1" phase (typically this is water), then the residual for the "2" phase is zero. This means the "2"-phase pressure will not change in those regions, potentially violating $P_1 \leq P_2$. If the "2" phase subsequently infiltrates to these regions, an initially crazy P_2 might affect the results. This sometimes also holds for almost-saturated situations, depending on the exact simulation.

In these cases, probably the best way of avoiding problems is to implement the constraint $P_1 \leq P_2$ using a `RichardsMultiphaseProblem` Problem object. In fact, I almost always use this as standard in my 2-phase simulations, just in case one phase disappears. This gets around the problem of chosing the a in the penalty term described in the next paragraph. See Section 5 for more comments.

Another way is to add a penalty term to the residual to ensure that $P_1 \leq P_2$. An example can be found in the tests directory `pressure_pulse/pp22.i`. The choice of the a parameter is sometimes difficult: too big and the penalty term dominates the Darcy flow; too small and the penalty term does nothing. In both cases, convergence is poor as the penalty term switches on and off during the Newton-Raphson procedure. The documentation for `RichardsPPenalty` describes how to set a (run MOOSE with a `--dump` flag).

The penalty term should *not* be used unless absolutely necessary as it will lead to poorer convergence characteristics. In many cases it is not necessary.

As mentioned above, in these situations it is often advantageous to use the fully-upwind kernels, boundary fluxes and dirac sources. If the immobile saturation of the phase is nonzero then it probably won't disappear entirely. If a phase disappears entirely then the Jacobian may be singular for $dt \to 0$, and this sometimes leads to nonconvergence. This can be avoided by using a nonzero residual saturation, or a shifted van Genuchten relationship. Please see Chapter 6 for more discussions and tips.

5 Bounding porepressure

Occasionally it might be useful to bound porepressure. The test `buckley_leverett/bl22.i` has "bounds" that do this. Note that:

- The convergence is likely to be much slower when using bounds
- The `-snes_type` must be set to `vinewtonssls` (see the `[Preconditioning]` block of the aforementioned test).
- The command line must contain the argument `--use-petsc-dm`.

The "bounds" just described uses PETSc to enforce $P \geq a$, for some fixed a. In two-phase situations, one often wants to enforce $P_1 \geq P_2$. To do this a `RichardsMultiphaseProblem` object can be used. See for example `gravity_head_2/gh_bounded_17.i`, which may be compared to `gravity_head_2/gh_lumped_17.i` to see how easy it is to use this in an input file. Note that:

- The convergence is likely to be much slower when using a `RichardsMultiphaseProblem` object.
- The use of this object will likely avoid crazy behaviour of a gas phase's porepressure when the phase is completely or almost completely absent in a region.
- Care must be taken: in the example `gh_bounded_17.i`, I had to use zero residual saturations, otherwise the simulation keeps trying to reduce `pgas` below `pwater` to conserve gas mass. If you see your nonlinear residual not decreasing, you might be using `RichardsMultiphaseProblem` to enforce a constraint which does not make physical sense (eg, it causes nonconservation of mass).

6 Singular Jacobians

In multiphase simulations, there are subtleties associated with one of the phases reducing to its residual saturation, and the resulting problems encountered can completely dominate the convergence characteristics of simulations. They can be understood by considering a simulation containing a single node, with no Darcy flux. The residual is just

$$R = \frac{d}{dt}\begin{pmatrix} \phi\rho_g S_g \\ \phi\rho_w S_w \end{pmatrix}. \tag{6.1}$$

Here I've labelled the phases "g" and "w", for gas and water. With the variables being P_g and P_w, the Jacobian is

$$J = \frac{\phi}{dt}\begin{pmatrix} \rho'_g S_g + \rho_g \frac{\partial S_g}{\partial P_g} & \rho_g \frac{\partial S_g}{\partial P_w} \\ \rho_w \frac{\partial S_w}{\partial P_g} & \rho'_w S_w + \rho_w \frac{\partial S_w}{\partial P_w} \end{pmatrix} = \frac{\phi}{dt}\begin{pmatrix} \rho'_g S_g + \rho_g S' & -\rho_g S' \\ -\rho_w S' & \rho'_w S_w + \rho_w S' \end{pmatrix} \tag{6.2}$$

Now imagine that $S_g = S_g^{\rm res}$ (that gas has reduced to the residual value). Using the standard van-Genuchten expression for S' gives

$$S' = 0 \tag{6.3}$$

and so

$$J = \frac{\phi}{dt}\begin{pmatrix} \rho'_g S_g^{\rm res} & 0 \\ 0 & \rho'_w(1 - S_g^{\rm res}) \end{pmatrix} \tag{6.4}$$

Note that if $S_g^{\rm res} = 0$, then this Jacobian is singular!

This singularity will manifest itself in various ways:

- When one phase disappears PETSc will find it difficult or impossible to invert the Jacobian, leading to a large number of linear iterations
- The nonlinear solver will find it difficult or impossible to converge.

The arguments above strongly point to ensuring that simulations are always run with

$$S^{\rm res} > 0. \tag{6.5}$$

for all phases. Then the Jacobian is non-singular if the timestep sizes are not too big, so that the time-derivative terms dominate the Richards flux terms. *However, this can also lead to difficulties.* Imagine that the 'g' phase is at its residual value. Then the difficulties encountered are:

1. If the 'w' phase wants to increase its pressure (because of Richards' flux, say), then the 'g' phase pressure *must increase* at least as fast as the 'w' phase to maintain $P_g \geq P_w$. However, if there is no 'g' phase entering the node (because of immobile saturation, say), mass conservation is thereby violated. This was mentioned in Chapter 5 in relation to the test `gh_bounded_17.i`. The nonlinear iterations in MOOSE will typically not converge in this case.

2. If the porosity decreases, and the 'g' phase is more compressible than the 'w' phase ($\rho'_g/\rho_g > \rho'_w/\rho_w$) the 'g' phase saturation will attempt to reduce below its residual value. In some ways this is unfair to Richards' flow, as the whole concept of residual saturation was never designed to handle dynamic porosity, but nevertheless this problem will manifest itself in nonconvergence of the nonlinear solver.

The existence of these two cases (which are not too unusual in real simulations) have made it necessary to create the "shifted" van-Genuchten capillary curve for two-phase situations.

The "shifted" van-Genuchten capillary curve is documented in the Theory and Test manuals. It has the important property that

$$S' > 0 \ , \tag{6.6}$$

for all finite porepressures. Therefore, the determinant of the Jacobian is always positive

$$\det J = \rho'_g \rho'_w S(1-S) + \rho'_g \rho_w S S' + \rho_g \rho'_w (1-S) S' > 0 \ . \tag{6.7}$$

since $0 \leq S \leq 1$, and $\rho' > 0$ physically, so the Jacobian is non-singular in the full Richards case for $dt \to 0$. (Here $S = S_g = 1 - S_w$.) Finally, using $S^{\rm res} = 0$ gets around both of the problems mentioned in the previous paragraph.