1. SIMULATION OF THE “SLOPING BASE AQUIFER” PROBLEM

The discharge of an unconfined double porosity aquifer with sloping base to an adjacent water stream was numerically simulated by COMSOL Multiphysics and OpenFOAM. Two simulators were used instead of a single one, because increased confidence in the results of the simulations was desired. The two simulators gave almost identical results.

* 1. COMSOL simulation (code in file “Slope.mph”)

Version 6.1 of COMSOL Multiphysics was used. COMSOL is a commercial general-purpose simulation software that uses the Finite Elements Method. For the problem at hand, the Model Wizard of COMSOL was used, providing a 1D model. Concerning the model Physics, the Mathematics toolkit of COMSOL was selected, and the “PDE Interfaces / General Form PDE” choice of the toolkit delivered a model of the general form

Here, with

The conservative flux term *Γ* was set equal to

The term in COMSOL stands for

The source term f was set equal to

Analytic functions were used to provide the initial values of and . Both initial values were set equal to

The Dirichlet boundary condition for at the water body was . The Neumann boundary condition for at the impervious boundary was set by

This boundary condition should result to Given the definition of above, the terms of this relation should be

Concerning the derivative of at , which is evaluated during the simulation over the entire field for all time values and is used for the evaluation of the specific discharge to the river, COMSOL offers the “Point Evaluation” functionality, which reports this derivative in a dedicated table.

This model was typically solved over a 1D field defined by 0 <= x\* <= 15 for positive angle θ=3ο, and by 0 <= x\* <= 40 for the other two cases, θ=0ο and θ=-3ο. The x\* field was shorter for positive angle because the initial values of and should stay positive, to prevent divergence of the algorithm. The dimensionless time variable t\* typically covered a range between t\*=0 and t\*=40. An isotropic mesh was used, with Δx\*=0.005. The increment of t\* was also constant, and equal to Δt\*=0.01, but in the first time steps a smaller value of Δt\*=0.001 was used. Notice that even with the values automatically suggested by COMSOL for Δx\* and Δt\*, which were much larger than the values used, the results were in most cases almost identical to the results obtained using the more accurate values of Δx\* and Δt\*. The remaining parameters of the COMSOL simulation were left at their default values.

* 1. OpenFOAM simulation (code in files “createFields.H” and “doublePorosityUnconfinedSlopeFoam.C”)

OpenFOAM is a free and open source simulator for Computational Fluid Dynamics. The simulator uses the Finite Volumes Method, and it is coded in C++. In the present work, version 8 of OpenFOAM from <https://openfoam.org/> was used (notice that there are several distributions of the software available). For the problem studied here, a custom solver “doublePorosityUnconfinedSlopeFoam” was developed and it is provided in appendices A1 (the variables definition, file “createFields.H”) and A2 (the solver code, file “doublePorosityUnconfinedSlopeFoam.C”). The solver is a variant of the PISO (Pressure Implicit with Splitting of Operators) solver, which solves the Navier-Stokes equations in unsteady problems. While in the COMSOL simulation the geometry of the field was 1D, in OpenFOAM a quasi-2D geometry with just a few cell rows (typically 5 to 10) along the vertical direction y was studied. The reason for this difference is that OpenFOAM works with 3D geometries, using similar tricks to emulate 3D fields for 2D or 1D problems. Further, the possible effect of different dimensionality in the two simulators was checked: this effect was found to be negligible, since both simulators provided almost identical fields. The additional dimensionless fracture head hf\* boundary conditions required at the top and bottom boundary of the quasi-2D field were set to “zero gradient” of hf\*.

Like in the COMSOL simulations, the initial values of and were set equal to with the funkySetFields command line utility, e.g., for an angle θ=3o, the command for was:

funkySetFields -time 0 -field hf -keepPatches -expression "1.0-0.052408\*pos().x"

The boundary conditions of the normalized heads were identical to those described above for the COMSOL simulations. The implementation of these conditions in OpenFOAM is straightforward.

In the OpenFOAM simulations, the sizes of the x\* and t\* ranges were the same with those used in the COMSOL simulations described earlier. Possible effect of the size of these ranges was also studied, but no effect of space and time ranges was noticed. Like in the COMSOL simulations, a time step of Δt\*=0.01 was used. In most simulations, the mesh was isotropic, i.e., all cells had the same size with a Δx\*=0.005 like in the COMSOL simulations. Several cases however were also ran with denser meshes around x\*=0, using the “grading factor” parameter of OpenFOAM (i.e., the ratio between the size of the end cell and the size of initial cell). A grading factor of 1.002 along x was used, but the results were identical to these obtained with isotropic mesh.

Concerning the derivative of at , notice the dedicated instruction for its recording in the solver code. Again, this derivative is always evaluated over the entire field.