Exercise Sheet No.6 Deadline: 3. Dez 2013

EXERCISE 6 A GROUNDWATER WELL

A hydraulic well which pumps water out of a confined aquifer can have a significant impact on the groundwater flow field in its surroundings. Such wells consist of a pump at ground level supplied by a pipe. The latter is perforated and henceforth assumed to be perfectly permeable to water.

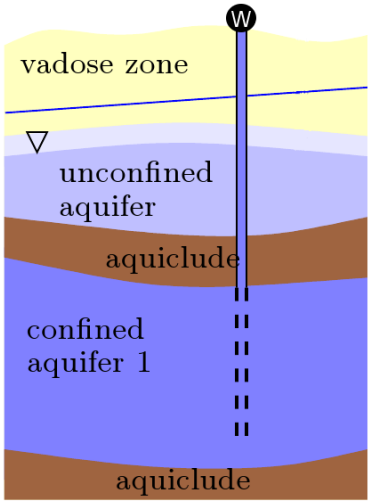

Figure 1: Sketch of a two story groundwater system.

The pump is defined by its total extraction rate $R\,[\frac{m^3}{s}]$, the drilling depth L within the confined aquifer and the pipe radius r. However, when developing a model for the pump, several possibilities arise:

1. The pump can be represented by the sink term $r_W(\vec{x})$ of the groundwater equation

$$-\bigtriangledown\cdot\left[\bar{K_S}(\vec{x})\cdot(\bigtriangledown p_W-\rho_W g\vec{e}_z)\right]+r_W\left(\vec{x}\right)=0.$$

   Then we have

$$r_W(\vec{x})=\begin{cases}-\dfrac{R}{Lr^2\pi} & \text{if } \vec{x} \text{ within the pipe}\\ 0 & \text{otherwise.}\end{cases}$$

2. As we expect the pump to extract more water at points with higher conductivity, it appears reasonable to adapt the sink term to the conductivity field $\bar{K_s}(\vec{x})$. In the simplest approach, this can be done by employing an arithmetic weighting:

$$r_W(\vec{x})=\begin{cases}-\dfrac{\bar{K_s}(\vec{x})\,R}{\int\limits_{Pipe}\bar{K_s}(\vec{x})dV} & \text{if } \vec{x} \text{ within the pipe}\\ 0 & \text{otherwise.}\end{cases}$$

3. If the resolution is high enough to resolve the pipe with acceptable error, another approach appears possible: For grid cells within the pipe, choose an extremely high conductivity and employ Neumann outflow boundary condtitions at the faces where the pipe meets the upper boundary:

$$\vec{J}_W=\frac{R}{\pi r^2}$$

In this exercise you will implement the groundwater well models given above and compare their performance. To allow a better visual evaluation of the simulation data, we will use the free open source software *paraview* developed by the Sandia National Labs, Kitware Inc, and Los Alamos National Labs (`http://en.wikipedia.org/wiki/Paraview`). The class `VTKWriter` is implemented in the file `vtkwriter.hh` which may be found on the lecture homepage. It allows the creation of VTK files that can be imported into paraview. Its interface consists of two public functions:

- `VTKWriter::VTKWriter(const Grid & grid)`

  Constructor which requires only the grid defining the geometry

- `void VTKWriter::write_groundwater(const std::string filename,`
  `const Vector & pressure,`
  `const SpatialParameters & conductivity,`
  `BoundaryConditions & boundary_conditions)`

  This functions creates a VTK file containing the conductivity field as given by `conductivity`, the pressure solution provided in `pressure` and the flow field. The latter is computed from the pressure solution and the boundary conditions given in `boundary_conditions`. The desired filename of the VTK file is to be given without file extension (`.vts` will be added automatically).

For all three of the well models given above, perform simulations on the following two-dimensional setup:

The confined aquifer is assumed to be rectangular with an extent of $50\,m \times 25.6\,m$. The conductivity field is given by the `cimg` file `field.cimg`. It defines a conductivity field $\bar{K}_s(\vec{x})$ of 512×256 pixels. Zero Neumann boundary conditions are assumed on the upper and lower boundary (impermeable aquiclude). On the left and right boundary we assume Dirichlet conditions. The pressure in the upper left corner is $10\,m$ and rises to $35.6\,m$ in the lower left corner. On the right boundary it rises from $0\,m$ to $25.6\,m$ (this compensates gravity and we achieve hydrodynamic equilibrium along both boundaries). The pipe penetrates the domain to a depth of $15.6\,m$ beginning at the center of the upper boundary. It is assumed to have negligible radius, hence its width is always chosen to be the horizontal grid resolution h_x. The total extraction rate of the well should be $R = 6 \cdot 10^{-4} \frac{m^2}{s}$ (remember that we are in the two dimensional world).

The class `BoundaryConditions` can provide the hydrodynamic equilibrium boundary condition if the type is define as `GravityDirichlet`:

```
boundary_conditions:
{
  # Define conditions depending on spatial range
  Spatial = (
  { type = "GravityDirichlet";
    xrange = "0";
    yrange = "0..25.6";
    zrange = "0..100";
    value = "35.6";
  },

  { type = "GravityDirichlet";
    xrange = "50";
    yrange = "0..25.6";
    zrange = "0..100";
    value = "25.6";
  }
  );

  # Default conditions
  Default = (
  { type = "Neumann";
    value = 0;
```

```
    }
    );
  };
```

The new boundary condition `GravityDirichlet` requires `value` to be the value of the pressure at the lower end of the specified boundary which is expected to be located at the zero coordinate of the gravity axis.

An appropriate software design for the realization of cases 1 and 2, would be to implement classes for the special source terms which inherit from `Sources` and provide the virtual method:
`double Sources::operator()(const double & position):`
For the realization of case 3 you should directly modify the cell values of the conductivity field. Therefore the method
`double & SpatialParameters::operator()(const size_t & id)`
may be used. You can use the file `well_models_incomplete.hh` as template.

Compare the performance of all three models qualitatively. Create appropriate visualizations of the corresponding flow and pressure field.

5 Points