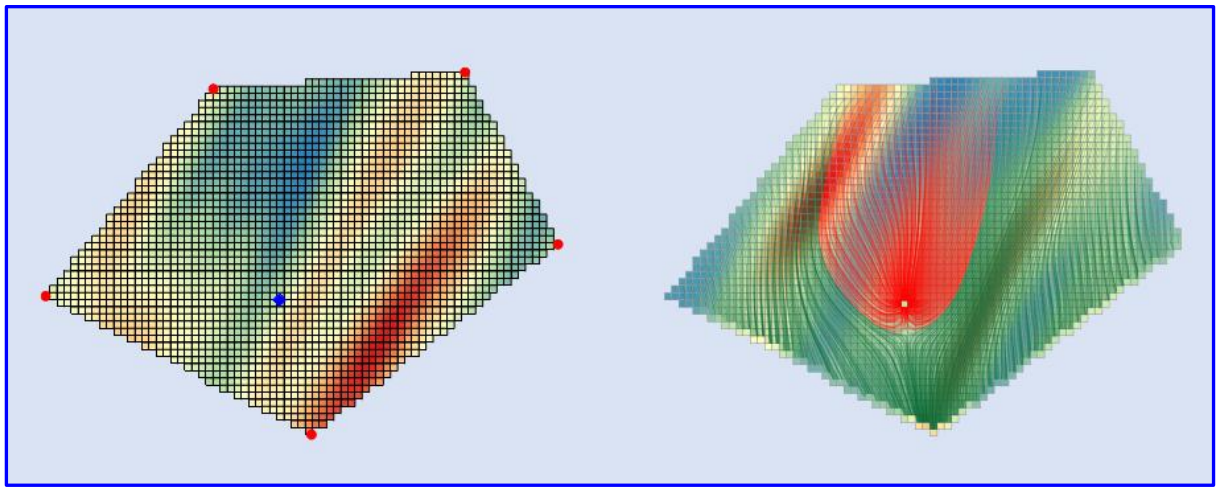
# Rapid Capture Zone Analysis
# for
# Springs and Wells



Watermark Numerical Computing

November, 2026

# Contents

1. Introduction ..... 1
2. Concepts ..... 2
2.1 The Model ..... 2
2.2 Hydraulic Properties ..... 3
2.3 Spring Flow ..... 3
2.4 Well Flow ..... 4
2.5 Capture Zone Definition ..... 4
2.6 Pictures ..... 5
3. Files ..... 6
3.1 General ..... 6
3.2 Input Control File ..... 6
3.2.1 Overview ..... 6
3.2.2 Model Boundary Heads ..... 9
3.2.3 Spring ..... 9
3.2.4 Well ..... 9
3.2.5 Background Hydraulic Properties ..... 10
3.2.6 Spatial Heterogeneity ..... 10
3.2.7 Spatial Correlation ..... 11
3.2.8 Control ..... 11
4. Individual Programs ..... 13
4.1 General ..... 13
4.2 BUILDMODEL ..... 13
4.3 GENPARAM ..... 14
4.4 MAKESHAPE ..... 15
4.5 RUNMODEL ..... 16
4.6 ADJPARAM ..... 18
4.7 CAPTUREPROB ..... 19
5. Workflow ..... 21
Appendix: An Extra Program ..... 22
A.1 PARTSHAPE ..... 22

# 1. Introduction

This manual documents a number of programs that collectively comprise a workflow for rapid, probabilistic, spring and well capture zone analysis. It also describes this workflow.

In the workflow described herein, definition of a spring or well capture zone is based on particle travel times. Movement of water and particles in the vicinity of a spring/well are simulated using the GWF and PRT models of the MODFLOW 6 simulator. Programs that are documented herein write input files for MODFLOW 6 and process MODFLOW 6 output files. MODFLOW 6 is run by the RUNMODEL program that is discussed below. The *mf6.exe* executable file is accessed through a *system()* call. Hence *mf6.exe* must reside in a folder that is cited in the PATH environment variable of a user's computer. Alternatively, it should reside in a user's working folder. It is important to ensure that *mf6.exe* pertains to the latest version of MODFLOW 6 is employed as some earlier versions had bugs in their simulation of particle movement.

Simulation of groundwater flow adjacent to a spring or well is somewhat abstract. This has the benefit of conceptual simplicity. It also enables fast model runs while facilitating rapid model calibration, post-processing and visualization. Moreover, because simulation of groundwater flow is numerical rather than analytical, representation of spatial heterogeneity can be enabled. Hopefully, errors that emerge from conceptual simplifications can be at least partly accommodated by stochastic representation of many aspects of the simulated system.

For analyses that are documented herein, a capture zone is defined as the area surrounding a spring or well in which groundwater is captured by the spring or well within a user-specified time. Groundwater residence times are evaluated over many stochastic realizations of local aquifer boundary conditions and local aquifer hydraulic properties. Capture probabilities are then assigned by assembling model outcomes based on these realizations.

# 2. Concepts

## 2.1 The Model

As stated above, capture zone analysis is undertaken using a MODFLOW 6 model whose domain encompasses a spring or well whose capture zone requires assessment. In order to define this model domain, a user must enclose the spring or well in a polygon. Ideally, the polygonal model boundary should be further from the spring/well than the dimensions of its anticipated capture zone. See Figure 1.

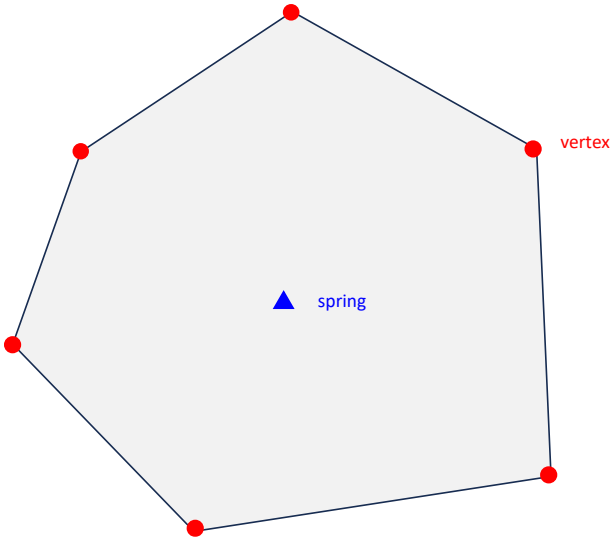



**Figure 1.1. A polygon surrounding a spring.**

A program named BUILDMODEL fills the user-specified polygon with a uniform grid. Rows of this grid are oriented in the east-west direction, while columns are oriented in the north-south direction. Because aquifer conceptualization is approximate, there is no need for this grid to be too fine, as this will slow stochastic hydraulic property field generation as well as simulation of groundwater flow and particle movement. It is suggested that 40 to 50 rows and columns are adequate.

As well as providing the locations of vertices that comprise the model boundary, a user must associate a head with each vertex. Because vertex heads are considered to be stochastic (in order to represent their uncertainties), mean values, standard deviations, as well as maximum and minimum values, must be provided for these vertex heads. Linear interpolation along polygon boundaries between vertex heads takes place for each stochastic realization of vertex heads. These linearly-interpolated heads populate a MODFLOW 6 CHD boundary that completely surrounds the spring or well.

The coordinates and elevation of the spring or well must also be provided by the user. A spring is simulated as an internal CHD boundary while a well is simulated as an internal DRN boundary. Both of these boundary types allow water to exit a model domain. However use of the DRN boundary allows simulation of flow between a cell which hosts a well and the well itself. The conductance of the drain is calculated using the Peaceman formula. This conductance is dependent on the diameter of the well, the dimensions of the enclosing model cell and the hydraulic conductivity of the medium in which the well is emplaced. It is calculated as:

$$C = \frac{2\pi Kt}{ln\left(\frac{r_0}{r_w}\right)} \qquad (2.1)$$

where:

*K* is the hydraulic conductivity assigned to the well-enclosing model cell;

*t* is the thickness of the aquifer;

$r_w$ is the radius of the well; and

$r_0$ is calculated using the grid cell size Δx as $r_0 = \Delta x e^{\frac{\pi}{2}}$ (2.2)

Care should be taken when assigning heads to vertices that surround a spring or well. These must be such as to allow the spring or well to flow. That is, water must escape from the model domain through the CHD or DRN boundary. Some trial and error, assisted by outputs of programs described below, may be required in setting these values. When simulating spring capture, BUILDMODEL will cease execution with an appropriate error message if at least one vertex head is not above the elevation of a spring. When simulating well capture, BUILDMODEL will cease execution if at least one vertex head is not above the well elevation minus maximum estimated well drawdown; see below. In many circumstances, some vertex heads will be greater than the spring or well-minus-drawdown elevation, while others will be lower, thereby collectively representing a hillside.

For computational simplicity, groundwater flow is simulated as confined and steady state. Recharge is ignored.

## 2.2 Hydraulic Properties

Both hydraulic conductivity and porosity can be represented as spatially distributed stochastic quantities. Software that is documented below (in particular the GENPARAM program), generates many realizations of these hydraulic property fields based on user-supplied geostatistical specifications. A user provides mean values for these hydraulic property types, as well as their standard deviations and spatial correlation lengths. The same correlation length is used for both hydraulic conductivity and porosity. However, geostatistical descriptors (often referred to as “hyperparameters”) such as mean, correlation length, together with anisotropy direction and magnitude, can themselves be characterized as uncertain, and can hence be represented stochastically. Capture zone estimation can therefore accommodate uncertainty in characterization of uncertainty.

As presently programmed, spatial correlation decreases with distance using a Gaussian decay function. Stochastic field generation is undertaken using the method of spatial averaging. This can be a little slow. However, it can readily accommodate nonstationary geostatistics in future versions of this software.

A user is given a wide range of choices pertaining to the nature and extent of hydraulic property heterogeneity that is introduced to the model domain. In some contexts, a user may wish to assume spatial uniformity of hydraulic conductivity and/or porosity, but desire that these spatially uniform properties be random. In other circumstances, spatial heterogeneity about random mean values may be the preferred option.

## 2.3 Spring Flow

As well as providing the elevation of a spring, a user must specify flow from the spring. This, too, can be expressed stochastically in order to reflect, for example, measurement errors and/or seasonal fluctuations of spring flow.

In order to ensure integrity of capture zone delineation, the model must reproduce measured spring flow. Few realizations of vertex heads and accompanying system hydraulic properties are likely to result in a calculated value of spring flow that replicates observed spring flow. Hence the model must be "calibrated".

Fortunately, because the calibration dataset is comprised of only one observation (i.e. spring flow), and because all boundary conditions are specified-head boundary condition, model calibration is an easy matter. It requires that domain-wide hydraulic conductivity be multiplied by a spatially uniform factor. Program ADJPARAM undertakes this multiplication. However, easy adjustment of hydraulic conductivity in this manner may not be possible for all realizations of vertex heads and concomitant spatial distributions of hydraulic properties as spring flow may not be positive for some of these realizations. (Note that positive spring flow is represented as negative CHD flow according to the MODFLOW protocol that flow of water is negative if it leaves a model domain.) Furthermore, a user must choose upper and lower limits for the hydraulic conductivity adjustment factor. A requirement for excessive hydraulic conductivity adjustment suggests that boundary heads and/or geostatistical hyperparameters should be reviewed.

## 2.4 Well Flow

As stated above, a well is represented using a DRN boundary condition, with the conductance of the drain accommodating flow from the well-containing model cell to the well itself. A user must ascribe a pumping rate to the well; this is assumed to be known and therefore possesses no uncertainty. The user must also ascribe a value to the drawdown in the pumped well; this value can be uncertain. Meanwhile, the elevation of the DRN boundary condition is set internally to the user-supplied top elevation of the well minus the well drawdown. In a process that is similar to that described above for spring flow, the ADJPARAM program adjusts hydraulic properties (within user-prescribed limits) in order for computed flow of water from the DRN boundary condition to equal the user-specified well pumping rate. If this cannot be achieved for a certain stochastic realization of hydraulic properties, then the hydraulic property realization is abandoned. If it can only be achieved for a small fraction of hydraulic property realizations, then geostatistical characterization of hydraulic properties should be revisited.

## 2.5 Capture Zone Definition

Once a suite of boundary head and hydraulic property realizations has been obtained that (after ADJPARAM adjustment) reproduce the measured spring flow or user-specified well extraction rate, the model can be run using each of these realizations with particles deployed to calculate groundwater residence times. Movement of particles is simulated using the MODFLOW 6 PRT model. A particle is placed at the centre of every active model cell. Particle travel times and termination locations are recorded for all stochastic realizations of model boundary heads and system hydraulic properties.

By processing these results, a probabilistic spring or well capture zone can be evaluated based on a user-prescribed threshold particle residence time. This is accomplished using program CAPTUREPROB. For each model cell, CAPTUREPROB counts the number of realizations for which a particle released in that cell reaches the spring or well in a time that is smaller than the user-specified threshold. (A travel time is declared to be infinite if a particle does not emerge at the spring/well at all.) This number is divided by the total number of realizations in order to evaluate the probability that the cell is within the spring or well capture zone.

This calculation is readily repeated for different user-specified residence time thresholds.

## 2.6 Pictures

Software that is documented below creates shapefiles that allow visualization of:

- hydraulic conductivity and porosity fields;
- a probabilistic capture zone;
- particle trajectories.

It is important for a user to inspect at least some realizations of hydraulic property fields. This allows him/her to feel confident that spatial heterogeneity is represented in the model to his/her satisfaction. If not, then geostatistical hyperparameters should be altered, and hydraulic property fields re-generated.

# 3. Files

## 3.1 General

Probabilistic evaluation of a spring or well capture zone requires the running of a number of programs in succession. All of these programs receive information about the files that they must read and write through their command line arguments. This enables them to be easily run from a graphical user interface or dashboard.

The first program that must be run during a capture zone workflow is BUILDMODEL. BUILDMODEL receives model and parameterization specifications from a user-prepared text control file. As is discussed in the following subsection, model and parametric specifications are associated with keywords; hence file preparation is easy. Furthermore, the keyword protocol facilitates introduction of new functionality to the program suite over time.

BUILDMODEL stores information that it calculates from the contents of its input control file in a binary "case file". This case file is read by other programs of the spring/well capture suite.

As discussed above, stochastic analysis requires generation and manipulation of many realizations of boundary vertex heads and system hydraulic properties. Because different hydraulic properties are assigned to different active cells of the model grid, the volume of numbers that must be transferred between executable programs that are described below can be large.

Numbers which comprise stochastic realizations of system hydraulic properties, as well as particle travel times that are calculated from these realizations, are independent of numbers that are used to specify how random fields are generated and how the MODFLOW 6 model is built. The latter numbers can be considered to comprise the "skeleton" of the capture zone simulation process. Numbers of this type include model cell sizes and dispositions, boundary locations, values of geostatistical hyperparameters and more. These numbers are stored in the binary case file that is written by BUILDMODEL. In contrast, the former numbers include realizations of hydraulic conductivity and porosity, as well as realizations of boundary vertex heads. Model run outcomes are added to this list of random variables as they are transferred between programs. All of these latter numbers are stored in a large CSV file. Each column of this file pertains to a single realization. Different members of the capture zone software suite that is described herein add rows to this CSV file, and/or process its contents.

## 3.2 Input Control File

### 3.2.1 Overview

Figure 3.1a shows an example of a BUILDMODEL input control file that is used for analysis of spring capture. Figure 3.1b shows an example of a BUILDMODEL input control file that is used for analysis of well capture.

```
START MODEL_BOUNDARY_HEADS

  START VERTICES
#     x          y          head_average  head_min  head_max head_stdev
    46411.695   770399.868   150.0         149.0     151.0    0.3
    48149.868   770515.746   155.0         154.0     156.0    0.2
    48791.655   769330.223   145.0         144.0     146.0    0.1
    47089.137   768019.908   140.0         138.0     142.0    0.5
    45252.912   768973.674   145.0         144.0     146.0    0.2
  END VERTICES
  HEAD_REL_SPRING no
END MODEL_BOUNDARY_HEADS

START SPRING
       SPRING_X      46866.294
       SPRING_Y      768946.933
       SPRING_ELEV   140.0
       FLOW_MEAN     250.0
       FLOW_MIN      230.0
       FLOW_MAX      260.0
       FLOW_STDEV    4.0
END SPRING

START BACKGROUND_HYDRAULIC_PROPERTIES
  K_BACK_MEAN        3.2
  K_BACK_LOGSTDEV    0.1
  K_BACK_MIN         3.0
  K_BACK_MAX         3.4
  POR_BACK_MEAN       .1
  POR_BACK_STDEV     .01
  POR_BACK_MIN       .05
  POR_BACK_MAX       .15
  THICKNESS         20.0
END BACKGROUND_HYDRAULIC_PROPERTIES

START SPATIAL_HETEROGENEITY
  K_MUL_LOGSTDEV     0.2
  K_MAX              20
  K_MIN              .1
  POR_ADD_STDEV      .04
  POR_MAX           0.2
  POR_MIN           0.01
END SPATIAL_HETEROGENEITY

START SPATIAL_CORRELATION
# Variable   mean   minval     maxval   distribution  stdev
  AHMAX      1000.0  800       1100     normal         50
  HANIS      2.0     1.0       3.0      uniform
  BEARING    30      25        35       uniform
END SPATIAL_CORRELATION

START CONTROL
  GRID_CELL_WIDTH  20.0
END CONTROL
```

**Figure 3.1a. A spring capture BUILDMODEL input control file.**

```
START MODEL_BOUNDARY_HEADS

  START VERTICES
#     x           y           head_average  head_min  head_max head_stdev
    46411.695   770399.868   150.0         149.0     151.0    0.3
    48149.868   770515.746   155.0         154.0     156.0    0.2
    48791.655   769330.223   145.0         144.0     146.0    0.1
    47089.137   768019.908   140.0         138.0     142.0    0.5
    45252.912   768973.674   145.0         144.0     146.0    0.2
  END VERTICES
  HEAD_REL_WELL no
END MODEL_BOUNDARY_HEADS

START WELL
       WELL_X           326053.0
       WELL_Y           4659148.0
       WELL_TOP_ELEV    470
       DRAWDOWN_MIN     8.0
       DRAWDOWN_MAX     12.0
       WELL_DIAMETER    0.01
       WELL_RATE        1000.0
END WELL

START BACKGROUND_HYDRAULIC_PROPERTIES
  K_BACK_MEAN         3.2
  K_BACK_LOGSTDEV     0.1
  K_BACK_MIN          3.0
  K_BACK_MAX          3.4
  POR_BACK_MEAN        .1
  POR_BACK_STDEV      .01
  POR_BACK_MIN        .05
  POR_BACK_MAX        .15
  THICKNESS          20.0
END BACKGROUND_HYDRAULIC_PROPERTIES

START SPATIAL_HETEROGENEITY
  K_MUL_LOGSTDEV      0.2
  K_MAX               20
  K_MIN               .1
  POR_ADD_STDEV       .04
  POR_MAX            0.2
  POR_MIN            0.01
END SPATIAL_HETEROGENEITY

START SPATIAL_CORRELATION
# Variable   mean   minval     maxval   distribution  stdev
  AHMAX      1000.0  800       1100     normal         50
  HANIS      2.0     1.0       3.0      uniform
  BEARING    30      25        35       uniform
END SPATIAL_CORRELATION

START CONTROL
  GRID_CELL_WIDTH  20.0
END CONTROL
```

**Figure 3.1b. A well capture BUILDMODEL input control file.**

As is apparent from the above examples, a BUILDMODEL input control file is subdivided into sections. These sections can be supplied in any order. Within each section, variables (or variable types in the case of the SPATIAL_CORRELATION section) are identified by keywords. Some keywords are optional; default values are supplied for omitted variables. Each section begins with a START statement and finishes with an END statement; the full name of the section must follow each of these statements.

A blank line can appear anywhere within a BUILDMODEL input control file. Any line that begins with "#" or "!" is treated as a comment line, and is therefore ignored.

A BUILDMODEL input control file must contain either a SPRING section or a WELL section. It cannot contain both of these sections.

The BUILDMODEL input control file is now discussed section by section.

### 3.2.2 Model Boundary Heads

The locations of vertices that define the polygonal model boundary are supplied within the VERTICES subblock of the MODEL_BOUNDARY_HEADS block. Each line within the VERTICES subblock must contain 6 entries. The first two entries are the *x* and *y* coordinates of a polygon vertex. Mean, minimum and maximum head values ascribed to the vertex follow these coordinates. The last entry on each line is the standard deviation of vertex head.

Note the following:

- BUILDMODEL will object if the minimum head exceeds either the maximum head or the mean head for any vertex. It will also object if the maximum head is less than the mean head or minimum head.
- To disallow stochastic representation of a vertex head, supply a head standard deviation of zero, or set the minimum, maximum and mean heads to the same value.
- Stochastic heads are represented using a truncated Gaussian (i.e. normal) distribution. It is the user's responsibility to ensure that the standard deviation of this distribution is commensurate with the difference between the maximum and minimum heads that are supplied for any vertex.
- Vertices within the VERTICES subblock should be provided in an order that circumnavigates the polygon; sides of the polygon that they define should not cross.

As stated above, vertices define a polygon that encloses a spring or well. If coordinates of the first vertex are not the same as those of the last vertex, BUILDMODEL closes the polygon itself. If coordinates of the first vertex are, indeed, the same as those of the last vertex but user-supplied values of head, minimum/maximum head, or standard deviation of head are not the same for the first and last vertices, then BUILDMODEL will cease execution with an appropriate error message.

Optionally, head elevations can be supplied relative to the elevation of the spring or well. If this is done, the value of the HEAD_REL_SPRING keyword, or the HEAD_REL_WELL keyword, should be set to "yes". The HEAD_REL_SPRING/HEAD_REL_WELL keyword is optional. If omitted, it is assumed to be "no".

Note that if HEAD_REL_SPRING is set to "yes", then at least one vertex head should be positive in order to allow the spring to flow.

### 3.2.3 Spring

Seven keywords should appear in the SPRING block. These are illustrated in Figure 3.1a. They are used to define the location of the spring as well as its elevation and flow. Flow is represented stochastically, using a truncated Gaussian distribution; mean, minimum and maximum values must be provided together with the standard deviation of spring flow. To suppress stochastic representation of spring flow, set the minimum flow value equal to the maximum flow value (which should also be equal to the mean flow value), or set the standard deviation of spring flow to zero.

### 3.2.4 Well

Seven keywords should appear in the WELL block. These are illustrated in Figure 3.1b. They are used to define the location of the well, its elevation, diameter and drawdown. As stated above, the

drawdown is that which is incurred within the well itself. Because this is unlikely to be known exactly, it is represented stochastically using a uniform distribution; minimum and maximum drawdown values must be provided. To suppress stochastic representation of well drawdown, set the minimum drawdown equal to the maximum drawdown.

The radius of the well is used in the Peaceman formula to calculate cell-to-well hydraulic conductance. If it is set too low, well drawdown may be calculated as too high. When providing a value for well diameter, a user should account for the thickness of the screen and gravel pack that surrounds the well intake.

### 3.2.5 Background Hydraulic Properties

If desired, spatial heterogeneity of hydraulic properties can be introduced to the model domain as perturbations about a background hydraulic property value; see below. Background values therefore provide mean values of spatially variable hydraulic properties. These mean/background values can themselves be stochastic. To preclude stochasticity, set the minimum value equal to the maximum value, or set the standard deviation of the background value to zero.

Background values can be provided for both hydraulic conductivity and porosity; see Figures 3.1a and 3.1b. Background hydraulic conductivity is assumed to follow a truncated log-normal distribution while background porosity is assumed to posses a truncated normal distribution. The standard deviation ascribed to background hydraulic conductivity should therefore pertain to the log (to base 10) of its value. However, K_BACK_MEAN, K_BACK_MIN and K_BACK_MAX should be expressed using natural values.

Aquifer THICKNESS must also be provided in the BACKGROUND_HYDRAULIC_PROPERTIES block. Although it is rarely known with precision, thickness is not represented stochastically in the workflow that is enabled by software that is described herein. Internally to MODFLOW 6, thickness is multiplied by hydraulic conductivity to obtain transmissivity. If a user is unsure of aquifer thickness, extra stochastic variability should be attributed to background hydraulic conductivity, or spatial variability of hydraulic conductivity, in order to accommodate uncertainty of aquifer thickness.

### 3.2.6 Spatial Heterogeneity

The SPATIAL_HETEROGENEITY block of the BUILDMODEL input control file is optional. If it is omitted, hydraulic conductivity and porosity are represented as uniform throughout the model domain. However, as is described above, this uniform value can be stochastic.

For hydraulic conductivity, a stochastic multiplier field is applied to the background field in order to represent spatial heterogeneity; this stochastic field has a mean value of 1.0. For porosity, a stochastic field is added to the background field; this stochastic field has a mean value of 0.0. The standard deviation of the log (to base 10) of the hydraulic conductivity multiplier field must be supplied following the K_MUL_LOGSTDEV keyword. The standard deviation of the porosity adder field must be supplied following the POR_ADD_STDEV keyword. Hydraulic property heterogeneity is assumed to be multiGaussian.

After undergoing multiplication or addition, hydraulic properties are truncated at values set by K_MAX and K_MIN (for hydraulic conductivity), and POR_MAX and POR_MIN for porosity. The user should ensure that these values are reasonably separated from each other in order to prevent large areas of hydraulic property constancy from appearing in otherwise heterogeneous, stochastic hydraulic property fields that are introduced to the model domain through the SPATIAL_HETEROGENEITY block.

## 3.2.7 Spatial Correlation

If a BUILDMODEL input control file contains a SPATIAL_HETEROGENEITY block, then it must also contain a SPATIAL_CORRELATION block. Spatial correlation is assumed to decay as the square of distance according to the formula:

$$c = k.e^{-\left(\frac{x}{a}\right)^2} \quad (3.1)$$

In equation (3.1), *k* is a constant of proportionality, while *x* is distance. *a* is roughly equal to the spatial correlation length.

BUILDMODEL allows correlation length to be a function of direction. That is, it allows anisotropic representation of spatial correlation. This can facilitate expression of heterogeneity associated with geological structure (which is not uncommon near springs).

The AHMAX variable that appears in the SPATIAL_CORRELATION block of the BUILDMODEL input control file is the value of *a* of equation 3.1 in the direction of maximum spatial correlation. This direction is set by the BEARING keyword; BEARING (in degrees) is measured clockwise from north. HANIS is the horizontal correlation anisotropy. This must be 1.0 or greater. A value of 1.0 indicates no anisotropy of spatial correlation. For values greater than 1.0, the value of *a* is set to AHMAX/HANIS (i.e. AHMAX divided by HANIS) in the direction perpendicular to BEARING.

All of the geostatistical hyperparameters that appear in the SPATIAL_CORRELATION block can be stochastic. As such, each can be ascribed a uniform probability distribution or a truncated Gaussian (i.e. normal) probability distribution. A value is required for standard deviation only if the probability distribution is decreed to be normal. As usual, set the maximum value equal to the mean and minimum values to eliminate stochastic expression of a geostatistical hyperparameter.

When assigning values to spatial correlation, it is wise to keep the dimensions of the model grid in mind. AHMAX should be significantly greater than the width of a model cell. The same applies to AHMAX/HANIS (i.e. spatial correlation in the direction of minimum correlation).

## 3.2.8 Control

A significant number of keywords can appear in the CONTROL block of a BUILDMODEL input control file. All of these keywords are optional; they are listed in Table 3.1.

| Keyword | Function | Default value |
|---|---|---|
| GRID_CELL_WIDTH | Width of a model grid cell. | See below |
| OUTER_DVCLOSE | MODFLOW 6 IMS variable | 1.0E-4 |
| OUTER_MAXIMUM | MODFLOW 6 IMS variable | 200 |
| INNER_DVCLOSE | MODFLOW 6 IMS variable | 1.0E-4 |
| INNER_RCLOSE | MODFLOW 6 IMS variable | 1.0E-4 |
| INNER_MAXIMUM | MODFLOW 6 IMS variable | 100 |

**Table 3.1 Keywords that can appear in the CONTROL block of a BUILDMODEL input control file.**

If the GRID_CELL_WIDTH keyword is omitted, BUILDMODEL calculates the size of model cells itself. It does this by ensuring that there are at least 50 active cells in the row and column directions of the model grid. All model cells are square.

A user should beware of setting GRID_CELL_WIDTH too low. This can increase model run times. It can also dramatically increase the time required for the GENPARAM program to generate stochastic hydraulic conductivity and porosity fields. GENPARAM uses the method of spatial averaging to

generate these fields. While this is a little slow, it is easily modified to allow spatial variability of geostatistical hyperparameters in future versions of this software suite.

As is apparent from the above table, the CONTROL block of a BUILDMODEL input file allows a user to provide values for some MODFLOW 6 solver settings. Changes to other MODFLOW solver settings can be made by directly editing the MODFLOW 6 IMS file that is written by BUILDMODEL.

# 4. Individual Programs

## 4.1 General

Individual programs are now described in the order in which they are generally used in a spring or well capture zone workflow. All programs receive their directives through command line arguments. Type the name of any program at the screen prompt to see its command line requirements.

## 4.2 BUILDMODEL

BUILDMODEL is run using the following command:

```
buildmodel infile casefile
```

where:

*infile* is the input control file, and

*casefile* is the binary case file that BUILDMODEL writes.

BUILDMODEL's input control file is described in the previous section of this document. *Casefile* is a binary file which summarizes (for other programs) all information that is provided in the BUILDMODEL input control file together with quantities that are calculated from this information.

As well as writing a binary case file, BUILDMODEL constructs a complete MODFLOW 6 input dataset. BUILDMODEL-generated MODFLOW input files are used by program RUNMODEL after some of these input files have been populated with stochastic fields. In contrast, hydraulic properties and boundary heads that appear in the MODFLOW 6 input dataset that is written by BUILDMODEL are mean/background values of these fields.

The MODFLOW 6 input dataset that is written by BUILDMODEL optionally allows the running of MODFLOW 6 with particles emplaced in every active cell of the model domain. Initially, however, the MODFLOW 6 PRT model should not be invoked, as early model runs are undertaken to calculate spring or well flows prior to adjustment of hydraulic conductivities in order to match these calculated flows to the observed flow. Simulation of particle movement can substantially increase model run times, especially if model cell sizes are small.

Program BUILDMODEL writes two *mfsim.nam* files. (MODFLOW 6 will not run unless a file of this name is present in the folder from which it is invoked.) One of these is named *mfsim.nam.part* while the other is named *mfsim.nam.nopart*. The latter is also recorded as *mfsim.nam*. Hence, after completion of BUILDMODEL execution, MODFLOW 6 can be run (without particles) by typing:

```
mf6
```

at the screen prompt. If you wish to conduct a model run using particles, copy *mfsim.nam.part* to *mfsim.nam* before running MODFLOW 6 in this manner.

Most of the MODFLOW 6 package input files that are written by BUILDMODEL have a root name of *springmodel* (when analyzing spring capture) or *wellmodel* (when analyzing well capture). However some of them (those pertaining to particle tracking) have root names of *springmodel_part* or *wellmodel_part*.

The input dataset that is written by MODFLOW 6 uses the OPEN/CLOSE protocol to access hydraulic conductivities and porosities from dedicated text files. These files are named *springmodel_k.ref* and *springmodel_por.ref* respectively when undertaking spring capture zone analysis, and *wellmodel_k.ref* and *wellmodel_por.ref* when undertaking well capture zone analysis. These files are overwritten on each occasion that MODFLOW 6 is run by the RUNMODEL program that is described below. So too is file *springmodel.chd* or *wellmodel.chd* that contains boundary heads. The same applied to file *springmodel_spring.chd* that represents a spring and file *wellmodel_well.drn* that represents a pumping well.

## 4.3 GENPARAM

As the name suggests, GENPARAM generates random realizations of stochastic variables that are used to populate the MODFLOW 6 model. However, GENPARAM does not run the model. Instead, stochastic realizations are stored in columns of a large CSV file that GENPARAM writes. Stochastic model outputs, calculated using stochastic model inputs, are appended to these columns by program RUNMODEL.

GENPARAM is run using the following command:

```
genparam casefile csvfile delfile nreal [iseed]
```

where:

| | |
|---|---|
| *casefile* | is the binary case file produced by BUILDMODEL; |
| *csvfile* | is the name of the CSV file that GENPARAM must write; |
| *delfile* | must be supplied as either “delcsv” or “nodelcsv”; |
| *nreal* | is the number of stochastic realizations that must be generated; and |
| *iseed* | is an optional seed for the random number generator. |

The *delfile* variable is included in the command line as a protective measure. As is documented below, a CSV file that is produced by GENPARAM may later be augmented with values of model outputs. A user may be disappointed if this file is inadvertently overwritten. If *delfile* is supplied as “nodelcsv”, then a CSV file that is named on the GENPARAM command line will not be over-written; instead GENPARAM will cease execution with an appropriate error message.

In the CSV file that is written by GENPARAM, stochastic values of polygon vertex heads are stored first. These are followed by cell-based values of hydraulic conductivity and then cell-based values of porosity. If analyzing well capture, the next row contains stochastic well drawdowns. For spring capture analysis, the last row of the GENPARAM-generated CSV file records values of observed spring flow; recall that observed spring flow can be represented stochastically in order to accommodate measurement errors and/or seasonal variability. Alternatively, if analyzing well capture, the last row of the GENPARAM-generated CSV file records the user-supplied well flow (i.e. the user-supplied well pumping rate). Each row of the CSV file is labelled (in its first column) for easy recognition of the contents of that row.

Obviously, if a model grid has many active cells, and if many realizations are requested, the CSV file that is written by GENPARAM may be large. Nevertheless, inspection and processing of its contents using programs such as EXCEL should generally be possible.

Note that realizations of porosity are generated independently of realizations of hydraulic conductivity. However, both of these property types share the same spatial correlation and anisotropy hyperparameters.

As stated above, the first column of a GENPARAM-produced CSV file provides names which identify the contents of each row. Meanwhile, the first row of this CSV file contains column headers in which realizations are numbered sequentially, starting at 1. Note that programs of the capture zone suite that are described herein require that realization numbers be in increasing order. Hence a GENPARAM-generated CSV file should not be re-arranged by the user. (The ADJPARAM program described below may remove some of these realizations, but will retain monotonicity of realization numbering.)

## 4.4 MAKESHAPE

Hydraulic conductivities and porosities that are listed in a GENPARAM-generated CSV file can be exported to a GIS using shapefiles that are written by MAKESHAPE. MAKESHAPE is run using the following command:

```
makeshape casefile csvfile proptype nreal1 nreal2 shapefilebase
```

where:

*casefile* is a binary case file produced by BUILDMODEL;

*csvfile* is a CSV file containing stochastic realizations of hydraulic conductivity and porosity produced by GENPARAM;

*proptype* should be supplied as “k” or “por” to indicate the hydraulic property type that will be recorded in the shapefile;

*nreal1* and *nreal2* are the numbers of the first and last realizations to be recorded in the shapefile; and

*shapefilebase* is the filename base of the shapefile that is written by MAKESHAPE.

Suppose that some columns have been removed from the GENPARAM-generated CSV file. (This can be done by the user or by program ADJPARAM.) Then GENPARAM internally adjusts *nreal1* to the first realization number that equals or exceeds the user-supplied value of *nreal1*; MAKESHAPE also internally adjusts *nreal2* to the last realization number that equals or is below the user-supplied value of *nreal2*. If a very high value is supplied for *nreal2*, then *nreal2* is reduced to the number that identifies the last realization that is recorded in the CSV file. (The numbers assigned to realizations are listed in the top row of the CSV file; as stated above, numbers will always be monotonic but may not always be sequential. *nreal1* and *nreal2* refer to these realization-identifying integers.)

In the shapefile that is written by GENPARAM, realizations of K are labelled as *krealN* where *N* is the realization number listed on the first row of the CSV file. Realizations of porosity are labelled *prealN*.

Figure 4.1 displays a realization of hydraulic conductivity imported into a GIS using a shapefile written by MAKESHAPE.

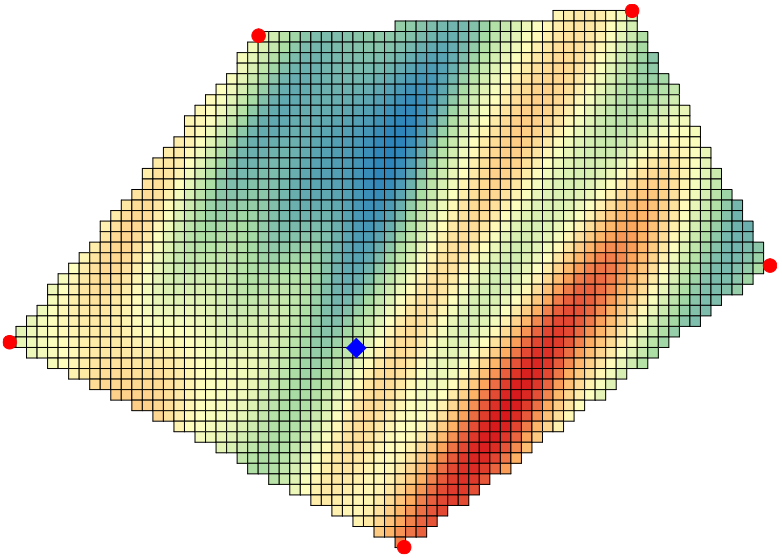


**Figure 4.1. A realization of hydraulic conductivity. User-supplied polygon vertices are shown in red while the location of a spring is shown in blue. (HANIS was set to 4 and BEARING was set to 30 for generation of this realization.)**

## 4.5 RUNMODEL

MODFLOW 6 can be run many times, using many realizations of stochastic hydraulic properties and boundary heads, using program RUNMODEL. Model outputs of interest are appended to the same CSV file as that from which RUNMODEL obtains stochastic values of vertex heads and hydraulic properties.

RUNMODEL is run using the following command:

```
runmodel casefile csvfile part [nreal1 nreal2]
```

where:

*casefile* is a binary case file produced by program BUILDMODEL;

*csvfile* is a CSV file containing stochastic vertex heads and stochastic hydraulic properties (produced by program GENPARAM or ADJPARAM);

*part* should be supplied as “part” or “nopart”; and

*nreal1* and *nreal2* are optional integer variables that identify realizations on which model runs should be based.

In the early stages of a capture zone workflow, *part* should be supplied as “nopart”, thereby requesting that model runs be undertaken without particles. The purpose of these model runs is to calculate spring flow or well flow using a suite of GENPARAM-generated stochastic fields. However if *part* is set to “part” then particles are emplaced at the centres of all active cells of the model grid. Program CAPTUREPROB can then be used to define a probabilistic capture zone. However, in general, capture zones should not be calculated using parameter fields for which calculated spring of well flows do not match the (stochastically enhanced) observed spring flow or the user-supplied well flow. Hence, in general, *part* should be set to “part” only if RUNMODEL is provided with a CSV file that was written by program ADJPARAM.

RUNMODEL records the outcomes of its calculations at the bottom of the CSV file that it reads to obtain realizations of vertex heads and hydraulic properties. Model-calculated spring or well flows are

recorded in the row that directly follows stochastically enhanced observed spring flows or the user specified well flow. If particles are employed, then subsequent rows of the CSV file are populated with cell-based particle travel times to the spring or well. A travel time of 1.1E30 indicates that a particle does not emerge from the groundwater system at the spring or well.

Note that, as stated above, rows of the CSV file are labelled (in its first column) for ease of recognition of row contents.

Suppose that values supplied for *nreal1* and *nreal2* are such that model runs are conducted for only a subset of realizations that are featured in a CSV file. (The numbers that identify realizations are listed in the top row of the CSV file; note that, for reasons explained elsewhere in this manual, these numbers are always monotonic but may not always be sequential.) Then RUNMODEL does not alter calculated values of spring/well flow and travel time contained in columns pertaining to other realizations. Alternatively, where no other runs have been previously undertaken, a dummy value of -1.1E30 is supplied for spring/well flow and a dummy value of 1.1E40 is supplied for travel time (if *part* is set to "part") for CSV columns that pertain to realizations for which model runs are not undertaken. These values can be subsequently over-written when RUNMODEL is used again to undertake further model runs based on realizations contained in other columns of the CSV file.

If *nreal1* and *nreal2* are omitted from the RUNMODEL command line, then the model is run for all realizations of boundary vertex heads and hydraulic properties that are featured in the CSV file that is supplied on its command line. (This is recommended practice.)

*Experience occasionally demonstrates that, when running MODFLOW 6 using particles, RUNMODEL ceases execution with the following error message displayed on the screen.*

```
MODFLOW runs in SEQUENTIAL mode

 Run start date and time (yyyy/mm/dd hh:mm:ss): 2026/07/21 12:00:05

 Writing simulation list file: mfsim.lst
 Using Simulation name file: mfsim.nam

    Solving:  Stress period:     1    Time step:     1
forrtl: severe (157): Program Exception - access violation
Image              PC                Routine            Line        Source
mf6.exe            00007FF788ED2948  PRTPRPMODULE_mp_I         557  prt-prp.f90
mf6.exe            00007FF788ECE60E  PRTPRPMODULE_mp_R         466  prt-prp.f90
mf6.exe            00007FF788ECD806  PRTPRPMODULE_mp_P         393  prt-prp.f90
mf6.exe            00007FF788C22EBB  PRTMODULE_mp_PRT_         380  prt.f90
mf6.exe            00007FF788CBC533  EXPLICITSOLUTIONM         238  ExplicitSolution.f90
mf6.exe            00007FF788CBC0C7  EXPLICITSOLUTIONM         206  ExplicitSolution.f90
mf6.exe            00007FF788B6021C  SOLUTIONGROUPMODU          83  SolutionGroup.f90
mf6.exe            00007FF788B22AD5  MF6COREMODULE_mp_         649  mf6core.f90
mf6.exe            00007FF788B21820  MF6COREMODULE_mp_          50  mf6core.f90
mf6.exe            00007FF788B21042  MAIN__                     11  mf6.f90
mf6.exe            00007FF78911C27E  Unknown               Unknown  Unknown
mf6.exe            00007FF78911C600  Unknown               Unknown  Unknown
KERNEL32.DLL       00007FFB3B18E957  Unknown               Unknown  Unknown
ntdll.dll          00007FFB3CECAD6C  Unknown               Unknown  Unknown
 Failed model run. Only 9 runs completed.
```

*The above error message is actually written by MODFLOW 6. It seems to be an outcome of the fact that MODFLOW 6 believes that it cannot write to its output particle-track-recording CSV file; this file is named springmodel_part.csv or wellmodel_part.csv. Immediately after MODFLOW 6 writes this file, it is read by RUNMODEL. Then, when MODFLOW 6 is run again using heads and hydraulic properties pertaining to the next realization, it appears that the operating system has not informed MODFLOW 6 that RUNMODEL has finished reading this file.*

*To alleviate this problem, RUNMODEL inserts a short delay between MODFLOW 6 runs that are undertaken for the purpose of particle tracking. Unfortunately, this lengthens RUNMODEL execution time. Furthermore, this does not always overcome the problem. So if it occurs, run RUNMODEL again. Or run RUNMODEL a number of times, using a subset of realizations on each occasion. If the problem persists, contact the programmer. Other measures can be adopted to prevent it.*

## 4.6 ADJPARAM

Program ADJPARAM is normally run following RUNMODEL when the latter is invoked to calculate spring or well flows using a suite of stochastic fields generated by GENPARAM. ADJPARAM writes a new CSV file in which all hydraulic conductivities within some realizations are multiplied or divided by a single number in order to ensure that model-calculated spring/well flows match the (stochastically enhanced) observed spring flow or the user-specified well flow. For reasons explained below, this may not be possible for all realizations. Hence the CSV file that is written by ADJPARAM often contains fewer columns than the CSV file that it reads. If an ADJPARAM-generated CSV file contains too few columns for reliable stochastic identification of a spring or well capture zone, this suggests that the capture zone workflow should be re-commenced (starting with BUILDMODEL) using re-defined stochastic hyperparameters.

ADJPARAM is run using the following command:

```
adjparam casefile incsvfile outcsvfile minadjfac maxadjfac
```

where:

*casefile* is the binary case file written by BUILDMODEL;

*inccsvfile* is an input CSV file (normally generated by GENPARAM);

*outcsvfile* is an output CSV file (normally with fewer columns than the input CSV file);

*minadjfac* is the minimum hydraulic conductivity adjustment factor to employ (1.0 or less); and

*maxadjfac* is the maximum hydraulic conductivity adjustment factor to employ (1.0 or greater).

The CSV file which ADJPARAM reads must have been produced by RUNMODEL so that it contains model-calculated spring or well flows in addition to (stochastically-enhanced) observed spring flows or the user-specified well flow. (The row that contains model-calculated spring/well flows immediately follows the row that contains observed spring/well flows.)

For each realization, ADJPARAM compares the observed spring or well flow to the modelled spring or well flow. Suppose that the modelled spring/well flow is less than the observed spring/well flow by a factor of *f*. Then ADJPARAM multiplies all hydraulic conductivities pertaining to this realization by *f*. Because all model boundaries are fixed head boundaries, this raises the model-calculated spring/well flow to a level that is equal to the observed spring/well flow. Similarly, if the model-calculated spring/well flow exceeds the observed spring/well flow, ADJPARAM reduces all hydraulic conductivities by an appropriate factor. Note that, in applying this factor, ADJPARAM does not clip hydraulic conductivities at values of KMAX and KMIN that are provided in the SPATIAL_HETEROGENEITY block of the BUILDMODEL input control file. (Linearity, and the ability to make a simple adjustment of hydraulic conductivity to equalize modelled and observed spring/well flows, would be lost under these circumstances.)

If, for a particular realization, the model-calculated spring or well flow is less than or equal to zero, this procedure will not work. Stochastic realizations for which this occurs are therefore not adjusted. Hence they are not transferred to the new CSV file that is written by ADJPARAM.

Also, if the ratio of an observed spring/well flow to a modelled spring/well flow is very large or very small, multiplication or division of an existing hydraulic conductivity field by a large factor may render some hydraulic conductivity values unrealistically large or unrealistically small. Hence the range of hydraulic conductivity adjustment factors should be limited to values for which this does not occur. This is done using the *minadjfac* and *maxadjfac* command-line arguments. The first of these must be supplied as 1.0 or less, while the second must be supplied as 1.0 or greater. If, for a particular realization, an adjustment factor outside this range is required for simulated-to-observed spring flow equalization, then ADJPARAM does not adjust this realization; nor is this realization transferred to the new CSV file.

In the new CSV file, each realization is given the same integer identifier as in the original CSV file. Hence, with omission of some realizations, numbering of realizations is no longer sequential. However ADJPARAM preserves monotonicity of realization numbering.

ADJPARAM reports to the screen the number of realizations that it transfers to the new CSV file. It transfers stochastic realizations of vertex heads, hydraulic conductivities, porosities, pumping well drawdowns and stochastically-enhanced, observed spring flows (or the user-specified well pumping rate) to the new CSV file. However, it does not transfer model-calculated spring/well flows or model-calculated particle travel times (if they are present) to the new CSV file as these are no longer model-calculated. Hence RUNMODEL must be run again in order to calculate model outputs that complete the new CSV file. Generally, however, after ADJPARAM adjustment of hydraulic conductivities, RUNMODEL should be instructed to use particles when running MODFLOW 6 so that spring or well capture probability can be calculated using boundary heads and hydraulic properties for which modelled and calculated spring/well flow rates have been equalized.

As already stated, if the number of realizations that are transferred to the new CSV file are insufficient for stochastic analysis, then parts of the preceding workflow may need to be repeated. Perhaps more stochastic fields should be generated using GENPARAM. Alternatively, perhaps statistical characterization of hydraulic properties provided in the BUILDMODEL control file should be adjusted; in the latter case, previous components of the workflow should be repeated in their entirety.

## 4.7 CAPTUREPROB

Once ADJPARAM has been run in order to produce a CSV file for which model-calculated spring/well flows match observed spring/well flows, MODELRUN should be invoked to calculate travel times for particles that are emplaced in all model cells. MODELRUN assigns a travel time of 1.1E30 to particles which do not terminate at the spring or well; the travel time for these particles is, therefore, effectively infinite. (After running MODELRUN, a user may wish to verify, by direct inspection of the CSV file that it writes, that measured and simulated spring/well flows are equal.)

The CSV file that is produced by MODELRUN in this manner is ready for use by CAPTUREPROB. CAPTUREPROB is run using the following command:

```
captureprob casefile csvfile timethresh arrayfile shapefilebase
```

where:

*casefile* is a binary case file produced by BUILDMODEL;

*csvfile* is a CSV file produced by MODELRUN with particle tracking enabled;

*timethresh* is the travel time threshold below which a cell lies within a capture zone;

*arrayfile* is a MODFLOW-compatible array containing cell capture probabilities; and

*shapefilebase* is the filename base for a shapefile containing capture probabilities ascribed to all active model cells.

For each cell, CAPTUREPROB evaluates the number of realizations for which a particle released in that cell reaches the spring or well with a travel time that is less than *timethresh* provided on its command line. It divides this number by the total number of realizations provided in its input CSV file to evaluate capture probability.

Capture probabilities that are stored in *arrayfile* can be readily imported into a MODFLOW graphical user interface. A capture probability is provided for all cells of the model domain whether they are active or inactive; naturally, inactive cells have a capture probability of zero. This file can also be processed using programs from the PEST Groundwater Utility suite.

Capture probabilities stored in a CAPTUREPROB-produced shapefile can be imported into a GIS for inspection. See below.

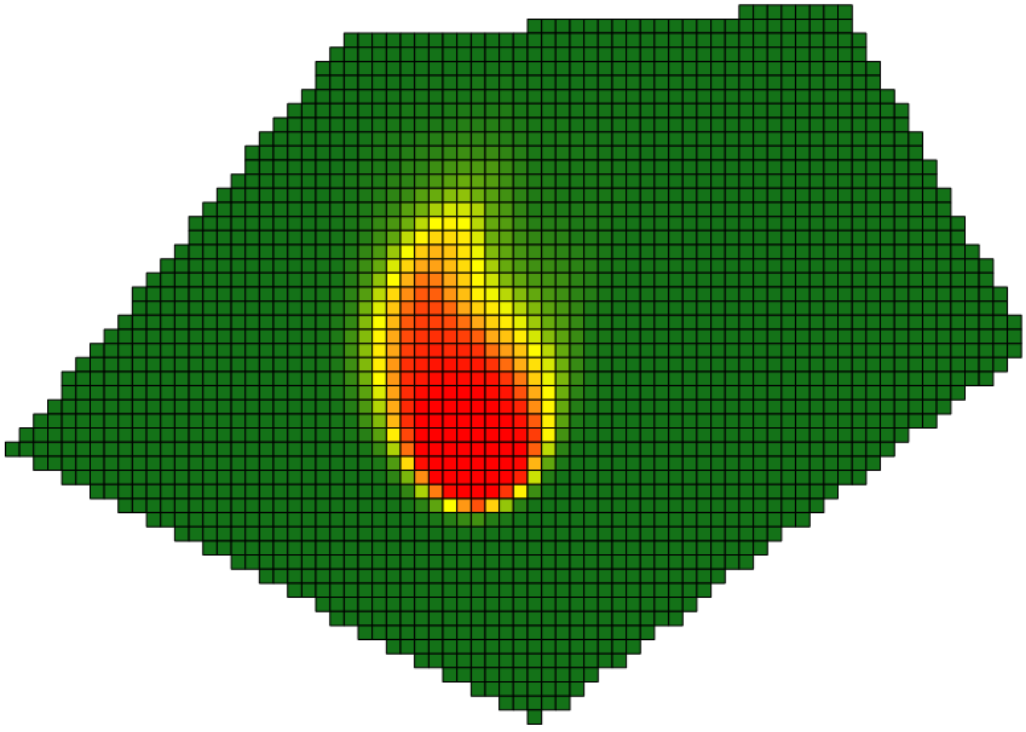


**Figure 4.2 Capture probabilities imported into a GIS.**

# 5. Workflow

The workflow for capture zone analysis should be readily apparent from the above descriptions of component software. It is summarized in the following flow chart.

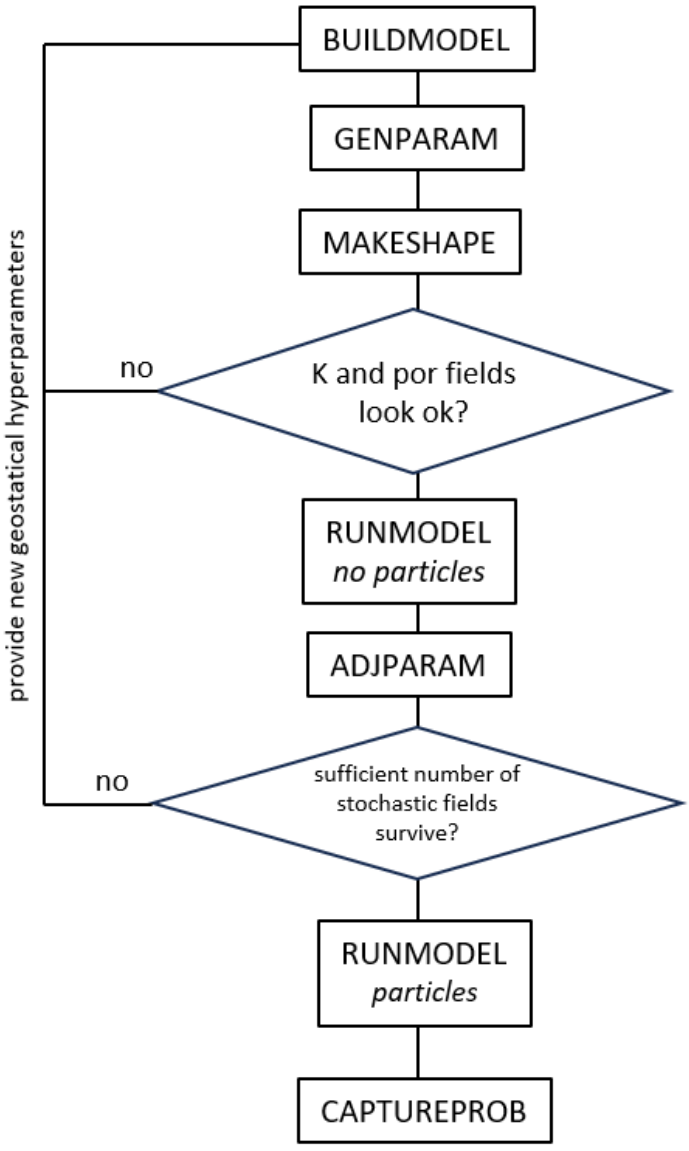



**Figure 5.1 Flow chart for spring or well capture zone probability analysis.**

# Appendix: An Extra Program

## A.1 PARTSHAPE

PARTSHAPE records a shapefile of particle tracks. It is documented in this appendix because it is not part of a capture zone workflow. However, its use may provide some insights into how water/particle capture is effected in the workflow that is documented herein.

PARTSHAPE is run using the following command:

```
partshape casefile shapefilebase
```

where:

*casefile* is a binary case file produced by BUILDMODEL; and

*shapefilebase* is the filename base for a shapefile containing particle tracks.

Movement of particles is simulated for many realizations of vertex head and hydraulic properties using MODFLOW 6 when run by the RUNMODEL program using the "part" command line option. Particle tracks are stored in a binary file named *springmodel_part.bin* or *wellmodel_part.bin* depending on whether spring or well capture zone analysis is being undertaken. PARTSHAPE reads the contents of this file. It knows its name because this is conveyed to it in the binary case file that is written by BUILDMODEL.

In the shapefile written by PARTSHAPE, particles that emerge at the spring or well are given an integer label of "1" while those that terminate in boundary cells are given an integer label of "0". As is shown below, this allows them to be displayed in different colours.

If PARTSHAPE is run at the end of the workflow that is described above, *springmodel_part.bin* or *wellmodel_part.bin* contains particles tracks calculated for the last realization contained in the CSV file on which RUNMODEL model runs were based. Alternatively, a user can select a particular realization for which to display particle tracks by running RUNMODEL with that particular realization selected on its command line before running PARTSHAPE.

An example of particle tracks, superimposed on a hydraulic conductivity field, is provided below.

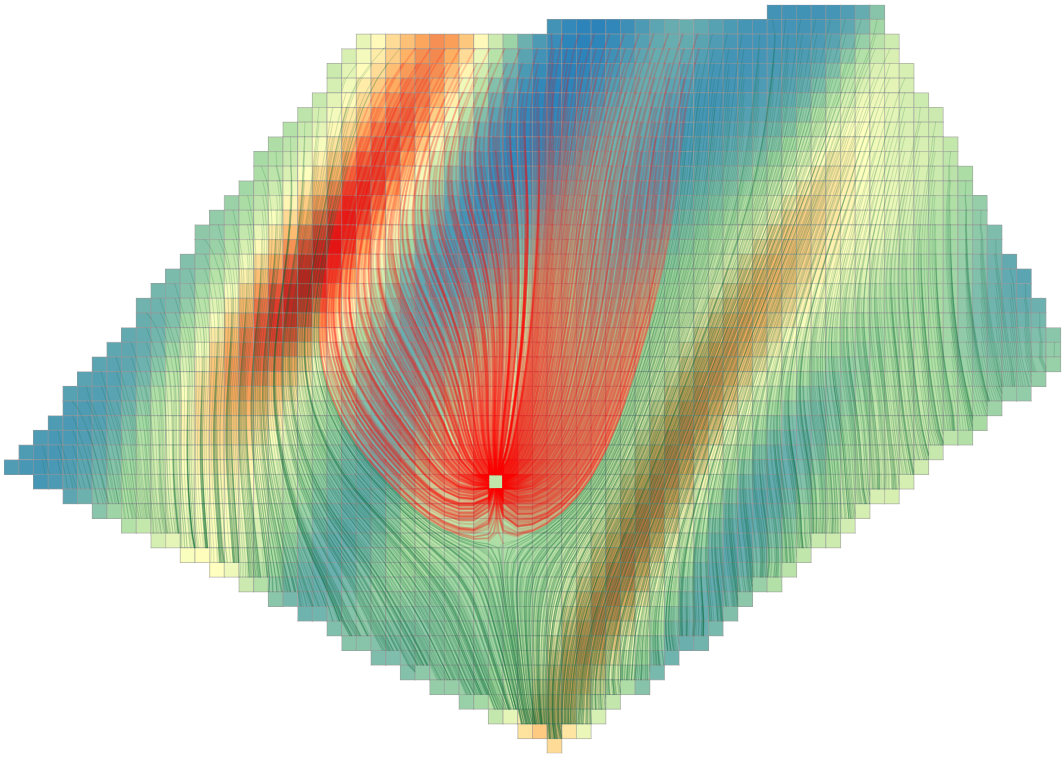


**Figure A.1 Particle tracks superimposed on a realization of hydraulic conductivity.**