## The PEST++ Command Line

PEST++ has a number of different command line options which are used to specify how to start a run using one of the supported run managers. Large problems (defined as having many parameters and/or observations) often require parallel computing. PEST++ relies on run mangers to complete the forward model runs and the current version provides the following three options: 1) **Y**et **A**nother Run **M**anage**R** (**YAMR**), 2) GENIE and 3) serial run manager. YAMR and GENIE are sophisticated and capable of performing parallel runs on a single machine or over a TCP/IP-enabled network. YAMR duplicates the functionality of BEOPEST and is fully integrated in PEST++. Like BEOPEST, PEST++ can run as either a master or a compute node under the YAMR run manager. While PEST++ provides an interface to the GENIE run manager, this interface relies on the external GMAN and GSLAVE programs to manage and perform the actual model runs. The serial rule manager provides a simple alternative that duplicates the functionality currently in regular PEST. The command lines required to start PEST++ are summarized in the following table.

|  |  |
| --- | --- |
| **Run Manger / Mode** | **Command** |
| Serial Run Manager / Master | pest++ control\_file.pst |
| YAMR / Master | pest++ control\_file.pst /H :port |
| YAMR / Compute Node | pest++ /H hostname:port |
| Genie / Master | pest++ control\_file.pst /G hostname:port |

## The PEST++ Control File

For ease of reference, variables within the PEST control file are listed below, and the variables used by PEST++ are highlighted. PEST++ relies on the structure of the input file to deduce the algorithmic parameters and read only those algorithmic parameters that are absolutely necessary. For example, there is no need to read the NOBS variable because each line in the “observation data” section of the control file specifies an observation; however, it is necessary to read the NPAR variable to know where specification of parameters ends and information on tied parameters begins. This list is followed by short explanation of each variable used by PEST++.

pcf

\* control data

RSTFLE PESTMODE

NPAR NOBS NPARGP NPRIOR NOBSGP [MAXCOMPDIM]

NTPLFLE NINSFLE PRECIS DPOINT [NUMCOM JACFILE MESSFILE]

RLAMBDA1 RLAMFAC PHIRATSUF PHIREDLAM NUMLAM [JACUPDATE] [LAMFORGIVE]

RELPARMAXFACPARMAXFACORIG [IBOUNDSTICK UPVECBEND] [ABSPARMAX]

PHIREDSWH [NOPTSWITCH] [SPLITSWH] [DOAUI] [DOSENREUSE]

NOPTMAXPHIREDSTPNPHISTPNPHINOREDRELPARSTPNRELPAR [PHISTOPTHRESH] [LASTRUN] [PHIABANDON]

ICOV ICOR IEIG [IRES] [JCOSAVE] [VERBOSEREC] [JCOSAVEITN] [REISAVEITN] [PARSAVEITN]

\* automatic user intervention

MAXAUI AUISTARTOPT NOAUIPHIRAT AUIRESTITN

AUISENSRAT AUIHOLDMAXCHG AUINUMFREE

AUIPHIRATSUF AUIPHIRATACCEPT NAUINOACCEPT

\* singular value decomposition

SVDMODE

MAXSINGEIGTHRESH

EIGWRITE

\* lsqr

LSQRMODE

LSQR\_ATOL LSQR\_BTOL LSQR\_CONLIM LSQR\_ITNLIM

LSQRWRITE

\* svd assist

BASEPESTFILE

BASEJACFILE

SVDA\_MULBPA SVDA\_SCALADJ SVDA\_EXTSUPER SVDA\_SUPDERCALC SVDA\_PAR\_EXCL

\* sensitivity reuse

SENRELTHRESH SENMAXREUSE

SENALLCALCINT SENPREDWEIGHT SENPIEXCLUDE

\* parameter groups

PARGPNMEINCTYPDERINCDERINCLBFORCENDERINCMULDERMTHD [SPLITTHRESH SPLITRELDIFF SPLITACTION]

(*one such line for each of NPARGP parameter groups*)

\* parameter data

PARNMEPARTRANSPARCHGLIMPARVAL1PARLBNDPARUBNDPARGPSCALEOFFSETDERCOM

(*one such line for each of NPAR parameters*)

PARNMEPARTIED

(*one such line for each tied parameter*)

\* observation groups

OBGNME [GTARG] [COVFLE]

(*one such line for each of NOBSGP observation group*)

\* observation data

OBSNMEOBSVALWEIGHTOBGNME

(*one such line for each of NOBS observations*)

\* derivatives command line

DERCOMLINE

EXTDERFLE

\* model command line

COMLINE

(*one such line for each of NUMCOM command lines*)

\* model input/output

TEMPFLEINFLE

(*one such line for each of NTPLFLE template files*)

INSFLEOUTFLE

(*one such line for each of NINSLFE instruction files*)

\* prior information

PILBL PIFAC \* PARNME + PIFAC \* log(PARNME) ... = PIVAL WEIGHT OBGNME

(*one such line for each of NPRIOR articles of prior information*)

\* predictive analysis

NPREDMAXMIN [PREDNOISE]

PD0 PD1 PD2

ABSPREDLAM RELPREDLAM INITSCHFAC MULSCHFAC NSEARCH

ABSPREDSWH RELPREDSWH

NPREDNORED ABSPREDSTP RELPREDSTP NPREDSTP

\* regularisation

PHIMLIM PHIMACCEPT [FRACPHIM] [MEMSAVE]

WFINIT WFMIN WFMAX [LINREG][REGCONTINUE]

WFFAC WFTOL IREGADJ [NOPTREGADJ REGWEIGHTRAT [REGSINGTHRESH]]

\* pareto

PARETO\_OBSGROUP

PARETO\_WTFAC\_START PARETO\_WTFAC\_FIN NUM\_WTFAC\_INC

NUM\_ITER\_START NUM\_ITER\_GEN NUM\_ITER\_FIN

ALT\_TERM

OBS\_TERM ABOVE\_OR\_BELOW OBS\_THRESH NUM\_ITER\_THRESH (*only if ALT\_TERM is non-zero*)

NOBS\_REPORT

OBS\_REPORT\_1 OBS\_REPORT\_2 OBS\_REPORT\_3..(*NOBS\_REPORT items*)

++# This line is a comment as are all lines that begin with “++#”

++# PEST++ input is parsed using key words that can be specified in any order

++ MAX\_N\_SUPER(max\_super) SUPER\_EIGTHRES(eig\_thres)

++ N\_ITER\_BASE(base\_iter) N\_ITER\_SUPER(super\_iter)

++ MAX\_FREEZE\_ITER(max\_frezze\_iter)

Variables in “control data” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| RSTFLE | Text | “restart” or “norestart” | Instructs PEST whether to write restart data. |
| PESTMODE | Text | “estimation”, “prediction”, “regularisation”, “pareto” | PEST’s mode of operation. |
| NPAR | Integer | greater than 0 | Number of parameters. |
| NUMCOM | Integer | optional; greater than zero | Number of command lines used to run model. |
| RELPARMAX | Real | greater than 0 | Parameter relative change limit. |
| FACPARMAX | Real | greater than 1 | Parameter factor change limit. |
| FACORIG | Real | between 0 and 1 | Minimum fraction of original parameter value in evaluating relative change. |
| PHIREDSWH | Real | between 0and 1 | Sets objective function change for introduction of central derivatives. |
| NOPTMAX | Integer | −2, −1, 0, or any number greater than 0 | Number of optimization iterations. |
| PHIREDSTP | Real | greater than 0 | Relative objective function reduction triggering termination. |
| NPHISTP | Integer | greater than 0 | Number of successive iterations over which PHIREDSTP applies. |
| NPHINORED | Integer | greater than 0 | Number of iterations since last drop in objective function to trigger termination. |
| RELPARSTP | Real | greater than 0 | Maximum relative parameter change triggering termination. |
| NRELPAR | Integer | greater than 0 | Number of successive iterations over which RELPARSTP applies. |

Variables in optional “singular value decomposition” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| MAXSING | Integer | greater than 0 | Number of singular values at which truncation occurs. |
| EIGTHRESH | Real | 0or greater, but less than 1 | Eigenvalue ratio threshold for truncation. |
| EIGWRITE | Integer | 0or 1 | Determines content of SVD output file. |

Variables required for each parameter group in “parameter groups” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| PARGPNME | Text | 12 characters or less | Parameter group name. |
| INCTYP | Text | “relative”, “absolute”, “rel\_to\_max” | Method by which parameter increments are calculated. |
| DERINC | Real | greater than 0 | Absolute or relative parameter increment. |
| DERINCLB | Real | 0or greater | Absolute lower bound of relative parameter increment. |
| FORCEN | Text | “switch”, “always\_2”, “always\_3”, “switch\_5”, “always\_5” | Determines whether central derivatives calculation is undertaken and whether three points or four points are employed in central derivatives calculation. |
| DERINCMUL | Real | greater than 0 | Derivative increment multiplier when undertaking central derivatives calculation. |
| DERMTHD | Text | “parabolic”, “outside\_pts”, “best\_fit”, “minvar”, “maxprec” | Method of central derivatives calculation. |

Variables required for each parameter in “parameter data” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| PARNME | Text | 12 characters or less | Parameter name. |
| PARTRANS | Text | “log”, “none”, “fixed”, “tied” | Parameter transformation. |
| PARCHGLIM | Text | “relative”, “factor”, or absolute(n) | Type of parameter change limit. |
| PARVAL1 | Real | any real number | Initial parameter value. |
| PARLBND | Real | less than or equal to PARVAL1 | Parameter lower bound. |
| PARUBND | Real | greater than or equal to PARVAL1 | Parameter upper bound. |
| PARGP | Text | 12 characters or less | Parameter group name. |
| SCALE | Real | any number other than 0 | Multiplication factor for parameter. |
| OFFSET | Real | any number | Number to add to parameter. |
| DERCOM | Integer | 0or greater | Model command line used in computing parameter increments. |
| PARTIED | Text | 12 characters or less | The name of the parameter to which another parameter is tied. |

Variables required for each observation group in “observation groups” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| OBGNME | Text | 12 characters or less | Observation group name. |

Variables required for each observation in “observation data” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| OBSNME | Text | 20 characters or less | Observation name. |
| OBSVAL | Real | any number | Measured value of observation. |
| WEIGHT | Real | 0or greater | Observation weight. |
| OBGNME | Text | 12 characters or less | Observation group to which observation assigned. |

Variables in “model command line” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| COMLINE | Text | system command | Command to run model. |

Variables in “model input/output” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| TEMPFLE | Text | a filename | Template file. |
| INFLE | Text | a filename | Model input file. |
| INSFLE | Text | a filename | Instruction file. |
| OUTFLE | Text | a filename | Model output file. |

Variables in “prior information” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| PILBL | Text | 20 characters or less | Name of prior information equation. |
| PIFAC | Text | real number other than 0 | Parameter value factor. |
| PARNME | Text | 12 characters or less | Parameter name. |
| PIVAL | Real | any number | “Observed value” of prior information. |
| WEIGHT | Real | 0or greater | Prior information weight. |
| OBGNME | Text | 12 characters or less | Observation group name. |

Variables in optional “regularization” section of PEST control file.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| PHIMLIM | Real | greater than 0 | Target measurement objective function. |
| PHIMACCEPT | Real | greater than PHIMLIM | Acceptable measurement objective function. |
| FRACPHIM | Real | optional; 0or greater, but less than 1 | Set target measurement objective function at this fraction of current measurement objective function. |
| MEMSAVE | Text | “memsave” or “nomemsave” | Activate conservation of memory at cost of execution speed and quantity of model output. |
| WFINIT | Real | greater than 0 | Initial regularization weight factor. |
| WFMIN | Real | greater than 0 | Minimum regularization weight factor. |
| WFMAX | Real | greater than WFMAX | Maximum regularization weight factor. |
| LINREG | Text | “linreg” or “nonlinreg” | Informs PEST that all regularization constraints are linear. |
| REGCONTINUE | Text | “continue” or “nocontinue” | Instructs PEST to continue minimizing regularization objective function even if measurement objective function is less than PHIMLIM. |
| WFFAC | Real | Greater than 1 | Regularization weight factor adjustment factor. |
| WFTOL | Real | Greater than 0 | Convergence criterion for regularization weight factor. |
| IREGADJ | integer | 0, 1, 2, 3, 4 or 5 | Instructs PEST to perform interregularizationgroup weight factor adjustment, or to compute new relative weights for regularization observations and prior information equations. |
| NOPTREGADJ | integer | 1 or greater | The optimization iteration interval for recalculation of regularization weights if IREGADJ is 4 or 5. |
| REGWEIGHTRAT | Real | absolute value of 1 or greater | The ratio of highest to lowest regularization weight; spread is logarithmic with null space projection if set negative. |
| REGSINGTHRESH | Real | less than 1 and greater than 0 | Singular value of **x**t**qx** (as factor of highest singular value) at which use ofhigher regularization weights commences if IREGADJ is set to 5. |

## PEST++ Additions to the PEST Control File

Information in the PEST control specific to PEST++ is specified on lines starting with “++”. Althoughthe previous example places all the PEST++ input in a single section at the end of the PEST control file, this is not a requirement.This information does not need to be contiguous and can reside anywhere in the PEST control file.Lines starting with “++#” are considered comments and are ignored.

Unlike the rest of the PEST control file, PEST++ uses keywords rather than location to specify variables.Lines are parsed using the space, tab, and parenthesis characters as separators.The example uses parentheses to more clearly delineate the values assigned to the variable, but these could just as well be replaced by white spaces.The following table includes a listing and explanation of the permissible PEST++ keywords.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Values | Description |
| GMAN\_SOCKET | Text | character string containing host and port separated by “:” | Socket of the GENIE GMAN run manager.The socket contains the hostname and port of the GMAN run manager that will be used to make the model runs.For example, if GMAN is running on the computer “my\_computer” listening to port 24772, then this variable should be specified asmy\_computer:24772. |
| N\_ITER\_BASE | Integer | 1 or greater | Number of base parameter iterations performed for each superparameter iteration. |
| N\_ITER\_SUPER | Integer | 0 or greater | Number of superparameter iterations performed for each base parameter iteration. |
| SUPER\_EIGTHRES | Real | any positive number (typically should be greater than 1.0e−7) | PEST++ will not include any superparameters whose ratio with the largest superparameter is less than this ratio.This value can as small as zero if the user wants to specify the number of superparameters solely with SUPER\_NMAX.BecausePEST++uses SVD on the superparameter problem, a low value for this SUPER\_EIGTHRES will not adversely impact the stability of the solution. |
| SUPER\_NMAX | Integer | integer between 1 and the minimum of maximum number of parameters and the maximum number of observations | Maximum number of superparameters to use in the superparameter iterations. |
| MAX\_FREEZE\_ITER | Integer | Interger between 1 and the maximum number of parameters that at expected to hit their bounds and need to be frozen (typically such be 1) | This option provides functionality similar the PEST’s “Automatic User Intervention” in that it freezes parameters one at a time recomputing the SVD transformation each step along the way. Use of this option can help a solution bouncing along its bounds converge, but can be computationally expensive for large problems. |

## References

Doherty, J., 2011a, PEST surface water utilities: Brisbane, Australia,Watermark Numerical Computing.

Doherty, J., 2011b, Groundwater data utilities: Brisbane, Australia, Watermark Numerical Computing.

Muffels, C.T., Schreüder, W.A., Doherty, J., Karanovic, M., Tonkin, M.J., Hunt, R.J., and Welter, D.E., 2011, GENIE—A model independent TCP/IP run managerinMODFLOW and More 2011—Integrated Hydrologic Modeling, Proceedings of the 10th International Conference of the International Ground Water Modeling Center: Golden, Colo., Colorado School of Mines.

Muffels, C.T., Schreüder, W.A., Doherty, J., Karanovic, M., Tonkin, M.J., Hunt, R.J., and Welter, D.E., 2012,Approaches in Highly Parameterized Inversion: GENIE, A General Model Independent TCP/IP Run Manager. U.S. Geological Survey Techniques and Methods Report **[UPDATE AFTER GENIE REPORT APPROVED].**

# Appendix 2: PEST++ Elementary Transformations

## TranScale

TranScale provides scaling or multiplication by a fixed value.The forward transformation is multiplication, and the reverse transformation is division.TranScale is one-to-one.

## TranOffset

TranOffset provides an offset or addition by a fixed value.The forward transformation is addition, and the reverse transformation is subtraction.TranOffset is one-to-one.

## TranFixed

TranFixed provides a transformation for fixing values.The forward transformation adds an additional parameter with a fixed value,and the reverse transformation removes the parameter.TranFixed is considered to be a one-to-one transformation becauseit is compatible with parameter back-substitution.

## TranFrozen

TranFrozen provides a transformation for freezing values.The forward transformation adds an additional parameter that is assigned the frozen value specified by this transformation, and the reverse transformation removes the parameter.The functionality of TranFrozen is identical to that of TranFixed.It has been implemented as an independent transformation for organization and tracking purposes.Like TranFixed, TranFrozen is considered to be a one-to-one transformation becauseit is compatible with parameter back-substitution.

## TranTied

TranTied provides a transformation that ties the value of one parameter to that of another parameter.The forward transformation adds an additional parameter that is assigned a value such that the ratio between the new parameter to the value of the parameter it is tied to is maintained at the level specified in transformation.The inverse transformation removes the tied parameter from the parameter set.TranTied is considered a one-to-one transformation becauseit is compatible with parameter back substitution.

## TranLog10

TranLog10 provides a base 10 logarithmic transformation.The forward transformation is the base 10 logarithm, and the inverse transformation is the exponential.TranLog10 is one-to-one.

## TranSVD

TranSVD provides a transformation between superparameters and base parameters.The forward transformation maps base parameters to superparameters, and the inverse transformation maps superparameters to base parameters.This transformation is used to implement the functionality of PEST’s SVD-Assist internally in PEST++.Additional details on SVD-Assist and this approach are provided in the next section.TranSVD is not one-to-one.

## TranNorm

TranNorm provides a transformation that automatically normalizes the parameters based on the assumption that the parameter range specified for each parameter in the input control file is indicative of its variance.The forward transformation divides each parameter by its variance, and the inverse transformation multiplies each parameter by it variance.TranNorm is one-to-one.

# Appendix 3: The Marquardt Lambda and SVD Rotation Factor Supporting Theory

The Marquardt lambda is typically a component of the Gauss-Marquardt-Levenberg method, but its use in PEST has been extended to supplement SVD.In the Gauss-Marquardt-Levenberg method, the Marquardt lambda is a weighting factor that interpolates between the Gauss-Newton method and the method of gradient descent.The Gauss-Newton solution for the nonlinear weighted least-squares problem can be written as

(1)

where is **u** the upgrade vector, **J** is the Jacobian, **Q** is the observation weights matrix and **r** is a vector containing the residuals of the observations.The matrix **JTQJ** in equation 1 is commonly referred to as the “normal equations matrix.”A full derivation is provided in Doherty (2010a).When the Marquardt lambda is added this equation becomes

(2)

where **I** is the identity matrix and is the Marquardt lambda.From equation 2, it is apparent that the Marquardt lambda adds terms to the diagonal of the normal equation matrix being inverted.Adding large terms to the diagonal of a matrix tends to make it better conditioned and provides a defacto form of regularization. However, this is not a desirable regularization strategy becauseit does not restrain the solution in a physically meaningful way—contrary to Tikhonov regularization (which identifies one or more preferred conditions) or truncated SVD( which minimizes changes for parameters whose influence on the inverse problem is overwhelmed by noise in the solution).So,althoughuse of the Marquardt lambda regularization is not recommended, it can nonetheless make an ill-posed problem solvable and is often unknowingly used in this capacity by new users.

Two of the Marquardt lambda’s roles have been discussed:rotating the solution vector in the direction of the gradient descent solution and providing a de facto form of regularization for ill-posed problems.In the authors’ view, it is also commonly used to add a more robust search capability to the nonlinear least-squares solution.Testing different values of the Marquardt lambda during each iteration allows PEST to explore a larger portion of parameter space, helping to prevent it from becoming trapped in a local minimum.When this technique is used in conjunction with parallel PEST, the impact on the overall runtime is minimalbecauseparallelization of testing parameter upgrades computed using different values of the Marquardt lambda allows these runs to be made with no increase in overall clock time.Because SVD incorporates subspace regularization, it does not benefit from the Marquardt lambda’s ability to provide suboptimal regularization; however,SVDcan greatly benefit from the addition of a more robust search capabilityto help avoid local optima, which is why the Marquardt lambda and SVD are commonly used together in PEST.

AlthoughPEST++ does not need to support the Gauss-Marquardt-Levenberg method, it still needs to provide an analogous functionality to a widelyvarying Marquardt lambda in the SVD solution.For large problems, performing an SVD factorization can be computationally expensive, and equation 2 requires that a SVD factorization be performed for each value of the Marquardt lambda becauseit is contained within the normal equation matrix being inverted.However, becauseSVD is unconditionally stable, the role of the Marquardt lambda can be reduced from adding a diagonal term to merely providing a means to rotate the upgrade vector in the direction of the gradient descent solution.To take advantage of this simplification, PEST++ replaces the Marquardt lambda with a newly defined rotation factor.The equation for the upgrade vector can be derived by using SVD and defining a unit vector pointing in the same direction to yieldto following equations:

(3)

(4)

where the symbol “-” denotes the generalized inverseand ||uSVD|| denotes the L2 norm of the upgrade vector. Similarly, the upgrade vector forthe direction method of gradient descent and its associated unit vector can be expressed as

(5)

(6)

Equations 4 and 6can be used to rotate equation 3 in the direction of the gradient descent solution to produce equation 7:

(7)

where is the rotation factor and u is the rotated upgrade vector.This technique avoids SVD factorization for each value of lambda, and its physical meaning is very clear.In addition, the rotation factor must lie in the interval [0, 1],whereasthere is no clear upper limit on Marquardt lambda.

# Appendix 4: Considerations for Code Development

## Design Goals

The initial goal of PEST++ development was to build a basic framework and create a program that makes the powerful features of PEST accessible to more users and developers while maintaining a robust and efficient design.In particular, the following items were identified as important.

1. Portability.—It is important that a well-established standard language be used and that the use of external dependencies be minimized.
2. Efficiency.—Becausethis program will be used for large problems, it must be efficient in two regards.First, it must be computationally efficient; and second, it must be efficient in its use of memory and maintain a “small memory footprint.”In particular, it must be possible to store and access nonstandard sparse information efficiently.
3. Extensibility.—The code must incorporate a modular or object-oriented design that promotes code extensibility and reuse.
4. Ease of use.—The code should automate processes and depend on default parameters when possible, thereby requiring as little user input as possible while providing the user with a simple interface for the required input data.Providing a more seamless interface to superparameters was identified as a key objective, given that userssometimes struggle with maintaining proper intermediate files and modified control file and batch files when using SVD-Assist in PEST.Also included in ease of use is support for robust error checking and handling.

## Language Selection

Selecting a language in which to build PEST++ was not obvious.During the pseudocode process, it became apparent that no singleprogramming languagewas optimum in all aspects for fully attaining the design goals.Much of the initial development of PEST++ was done in Python to take advantage of its concise syntax and friendly development environment.Use of Pythonat the outset allowed for rapid development and testing of an initial prototype. However, it became apparent that handlingmany of the sparse data structures efficiently would be difficult ifusing Python’s standard data types and packages.To avoid these issues and ensurethat adequate options will be available to resolve performance issues as they arise, the code was migrated to C++.C++ was chosen for this project becauseit offers the following benefits:(1) It is a mature language, governed by ISO standards.(2) It is widely available on many different platforms.(3) The standard library thatis included in the standard language provides a rich set of tools, including the standard template library (STL).(4) C++ includes exception handling, which can handle usage errors in an efficient manner.(5) An experienced programmer can develop fast and efficient code.(6) It is relatively easy to produce statically linked executables, which are easy to distribute.

External PEST++ dependencies have been limited to BLAS, LAPACK, LAPACK++ and PROPACK, where LAPACK++ is an object-oriented wrapper for the BLAS and LAPACK linear algebra libraries and PROPACK is an iterative solver for computing SVD factorizations.

## Integration in Visual Studio Integrated Development Environment(IDE)

All the source code required to build PEST++ has been consolidated in the Microsoft Visual Studio 2010 IDE.This includes the source codes for(1) PEST++,(2) the GENIE interface,(3) LAPACK++, and (4) the PEST Fortran code that processes template and insertion files.The project is configured to build a statically linked PEST++ executable without any external dependencies.This simplifies and facilitates sharing the source code and distributing the executables.The PEST++ Visual Studio Project, as well as souce code and executable, are available for download at: *www.pesthomepage.org*.

## Naming Convention

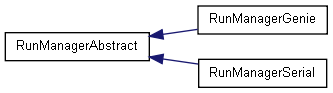
The naming convention used for PEST++ code is summarized in table 4–1 and follows that of the standard template library.AlthoughPEST++ code follows this convention, the LAPACK++ library used by PEST++ adheres to a different naming convention.

**Table 4–1.** PEST++ naming convention.

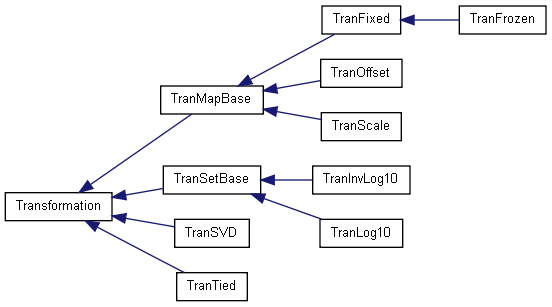
|  |  |  |
| --- | --- | --- |
| Items | Convention | Example |
| Class names | Camel-case.Names start with capitals and new words are delineated by capitals. | Class MyClass |
| Class instances  Class methods  Variables  Functions | Names start with lower case and boundaries are delineated with “\_” | MyClass my\_class  void MyClass.do\_somenting(int a)  double my\_value  int my\_function(int a) |

## Inheritance and Polymorphism in PEST++

Polymorphism is an object-orientated programming feature that allows data types sharing a common interface to be used interchangeably within a program.PEST++ uses polymorphism based on inheritance to define interfaces for run mangers, elemental transformations, and SVD solutions. In the case of the run manger, this shields the PEST++ code from having to specifywhich run manger is being used and makes the code compatible with any run manger that conforms to the ModelRunManagerAbstract base class.Figures 4–1, 4–2, and 4–3 show the inheritance trees for the run manager, elementary transformations, and the SVD-based solution.



**Figure 4–1.** Run manager inheritance tree.



**Figure 4–2.** Elemental transformation inheritance tree.

## C:\Users\dwelter\Documents\Visual Studio 2010\Projects\pest++\pest++\doxygen\html\inherit__graph__31.png

**Figure 4–3.** SVD Solution inheritance tree.

## Overall Program Flow and Design

The following flowcharts (figs. 4–4 through 4–6) describe the overall flow of PEST++,SVDSolver::solve, and SVDSolver::iteration.

Cleanup

andRelease Memory

Build Data Structures

Base Parameter Iteration(s)

Call SVDSolver::solve

Superparameter Iteration(s)

Call SVDASolver::solve

Iteration Criteria

Satisfied?

Main PEST++ Program

Read Control File

yes

no

**Figure 4–4.** Flowchart for main PEST++ program.

Return

Call SVDSolver::iteration

Iteration Criteria

Satisfied?

yes

no

SVDSolver::solve

**Figure 4–5.** Flowchart for SVDSolver::solve.

Calculate Jacobian

Call Jacobain::calculate

Compute Upgrade Vectors Based on Newton’s Method and Method of Gradient Descent

Use Rotation Factors to Compute Upgrade Vectors and Build the Associated Model Runs

Test Upgrade Vectors – Run Model

Save Updated Parameters and Jacobian

Freeze Parameters as Necessary

New Parameters Frozen?

yes

no

SVDSolver::iteration

**Figure 4–6.** Flowchart forSVDSolver::iteration.

# Appendix 5: Class List

## [FileManager](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_file_manager.html)

This class encapsulates the I/O filenames and iostream handles associated with a PEST++ run.It has a copy of the pathname in which the PEST++ simulation is running and a reference to the file stream associated with the record file.The rest of the class in PEST++ program relies on this class to supply the iostream handle to the record file, as well as the appropriate names of all PEST++ input and output files.

## [Jacobian](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_jacobian.html)

This class provides support for building and accessing the Jacobian.It includes code for calculating derivatives and provides support for forward difference, central difference(outer), and central difference(parabolic), as wellas the ability to switch to from forward derivatives to central derivatives as the optimization process slows down.This class uses the GENIEinterface to perform the actual model runs and isable to return the Jacobian as a LAPACK++ LaGenMatDouble matrix thatcan be used in numerical computations and save the Jacobian to disk PEST’s .jco data format.

## [ModelExecInfo](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_exec_info.html)

Thisclass contains the commands and filenames required to make a model run.It encapsulates the names of the template files, input files, insertion files, and output files. In addition, it also contains the model command lines necessary to start a model runs.

## [ModelRun](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run.html)

ModelRun is a child of [ModelRunAstractBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run_astract_base.html) and is designed to consolidate the information associated with a model run.Because[ModelRunAstractBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run_astract_base.html) is an abstract class, it cannot be directly instantiated.ModelRun is the primary child class that is used most often to store the parameter and simulated observations associated with a model run.

## [ModelRunAstractBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run_astract_base.html)

This is the abstract base class for all classes associated with a model run.These classes are designed to consolidate the information associated with a model run, which includes the parameters, transformations, and the observed values, as well as their simulated counterparts.In addition, these classes also contain the information necessary to compute the objective function and contain methods thatperform this task.Although[ModelRunAstractBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run_astract_base.html) cannot be instantiated, it defines the methods that all classes derived from it must implement.

## [ModelRunShallowCopy](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_model_run_shallow_copy.html)

This is a lightweight copy of a ModelRun class.It contains its own set of parameters and simulated observations, but it uses the transformations associated with the ModelRun class it was created from.

## [ObjectiveFunc](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_objective_func.html)

This class handles all aspects of the objective function.It contains the measured observations and prior information, along with all the associated in information such as weights and groups, and is able to calculate the objective function and produce a report thatincludes the contributions to the objective function of all of all the observation groups.

## [ObservationGroupRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_observation_group_rec.html)

This class stores the information associated with an observation group.

## [ObservationInfo](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_observation_info.html)

This class compliments the Observations class.Becausethe Observations class was designed to be very lightweight, it only stores the measured values of the observation.The ObservationInfoclass uses an unorder\_map of the ObservationRec class to store the rest of the information associated with the observations, which includes their weights and the groups that they are associated with.

## [ObservationRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_observation_rec.html)

This class stores the weight and observation groups associated with an observation.

## [Observations](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_observations.html)

This class stores the measured values of the observations.It is a child of the Transformable class and is compatible the transformations used for the parameters, but the current code does not make use of this functionality.

## [OperSys](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_oper_sys.html)

This class encapsulates all the features and parameters that are operating-system dependent.These include the character used to separate the different components of a pathname and the end-of-line or carriage-return character(s).

## [ParameterGroupInfo](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameter_group_info.html)

This class stores the information associated with the parameter groups.

## [ParameterGroupRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameter_group_rec.html)

This class stores the information associated with a parameter group.

## [ParameterInfo](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameter_info.html)

This class compliments the [Parameters](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameters.html) class.Because the [Parameters](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameters.html)class was designed to be lightweight, it stores only the parameter values.The ParameterInvoclass uses anunorder\_map of the [ParameterRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameter_rec.html)class to store the rest of the information associated with parameter, which includes their bounds, change limit methodology, and the groups they are associated with.

## [ParameterRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameter_rec.html)

This class stores the bounds, change limit methodology, and the group for a parameter.

## [Parameters](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_parameters.html)

This class stores parameter values.It is a child of the Transformable class and is designed be compatible the [ParamTransformSeq](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_param_transform_seq.html) class.

## [ParamTransformSeq](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_param_transform_seq.html)

This class handles transformations for the parameters.It contains the transformation sequences to convert between control and numeric parameters, as well as those required to convert between control and model parameters.

## [Pest](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_pest.html)

This class consolidates the information contained in the control file and contains instances of the following classes: ControlInfo, SVDInfo, Parameters, ParameterInfo, ParameterGroupInfo, BaseGroupInfo, Observations, ObservationInfo, PriorInformation, ModelExecInfo, PestppOptions, and ParamTransformSeq.

## [PestConversionError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\classpest__conversion__error.html), [PestError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\classpest__error.html), [PestFileError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\classpest__file__error.html), [PestIndexError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\classpest__index__error.html), [PestParsingError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\classpest__parsing__error.html)

These classes define error states and are used to throw exceptions.

## [PestppOptions](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_pestpp_options.html)

This class stores the PEST++ input options that are not available in PEST.

## [PIAtom](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_p_i_atom.html)

This class is used by the [PriorInformationRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_prior_information_rec.html) class.It stores information associate with a single parameter in a prior information expression.This information includes the name of the parameter, whether the parameter is log transformed, and the factor by which the parameter is to be multiplied in the prior information expression.

## [PriorInformation](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_prior_information.html)

This class contains all for the prior information records associated with a PEST++ simulation.

## [PriorInformationRec](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_prior_information_rec.html)

This class stores a complete prior information record and contains a method to compute that item’s residual and contribution to the objective function.Each prior information equation in the control file is stored by using an instance of this class.

## [QSqrtMatrix](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_q_sqrt_matrix.html)

This class contains the weights matrix for the observations and the prior information.It contains methods thatreturn the product of this matrix with a LAPACK++ LaGenMatDouble matrix.

## [RunManagerGenie](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_run_manager_genie.html)

This class is a wrapper for the GENIE run manager.GENIE is a generic parallel run manager that uses TCP/IP for communication and allows runs to be performed over the Internet in remote locations.This class converts PEST++ data structures to GENIE data structures and calls GENIE to make the model runs.

## [SVDASolver](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_s_v_d_a_solver.html)

This class is a child of the SVDSolver class that is specialized to work with superparameters.The changes needed to accommodate superparameters are minimal; the only methods that are modified are the method that limits parameter upgrades, limit\_parameters\_ip, and the method that freezes parameters, freeze\_parameters.

## [SVDInfo](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_s_v_d_info.html)

This class stores the information in the SVD section of the control file.

## [SVDSolver](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_s_v_d_solver.html)

This class solves the least-squares problem by using singular value decomposition (SVD).The bulk of the work is performed in the iteration method, which setsup and solves a single iteration.This class relies on the Jacobian class to compute the Jacobian.

## [TerminationController](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_termination_controller.html)

This class manages the termination criteria for the least-squares solution.

## [TranFixed](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_fixed.html)

This class implements a transformation for fixed values.

## [TranFrozen](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_frozen.html)

This class implements a transformation for frozen values.

## [TranLog10](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_log10.html)

This class implements a logarithmic transformation.

## [TranMapBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_map_base.html)

This is a base class for transformations built around the standard template library (STL) map class.

## [TranOffset](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_offset.html)

This class implements the offset transformation.

## [TranScale](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_scale.html)

This class implements a scaling transformation.

## [TranSetBase](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_set_base.html)

This is a base class for transformation that use the STL Set container to store a list of items.

## [Transformable](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_transformable.html)

This is an abstract base class thatprovides compatibility with the Transformation class and all of the class derived from it.The Parameters and Observation classes are derived from this class; however, at present, only the Parameters class makes use ofthe functionality it provides.

## [TransformableValueError](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_transformable__value__error.html)

This class definesan error state and is used to throw an exception.

## [Transformation](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_transformation.html)

This is the abstract base class for all transformations.It cannot be instantiated, but it defines the methods that all of its children must possess.

## [TranSVD](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_s_v_d.html)

The Transformation implements SVD-A or a transformation between superparameters and base parameters.

## [TranTied](file:///C:\Users\dwelter\Documents\Visual%20Studio%202010\Projects\pestalito\pestalito\doxygen\html\class_tran_tied.html)

This transformation ties one parameter to another.The tiedparameter is maintained at a fixed ratio to the parameter it is tied to.

# Appendix 6: Simple Storage Model Example

## Introduction

### This example demonstrates the use of PEST++ with a simple storage modeldeveloped by John Doherty as a workshop problem for use in PEST training classes.The brief description contained herein is taken from that exercise.The outputs of both PEST and PEST++ are presented so that users can review and verify the results.

## Description of the Model

Figure 6–1 shows a simple storage.

### 

**Figure 6–1.** A storage filled with a porous medium.

The storage is filled with a porous medium of storage coefficient *S*. The storage receives water as recharge at a constant rate *R*. Water is able to drain from the storage at a rate thatis proportional to the head of water in the storage. Rate of water outflow is thus given by

##### q = Kh (1)

where *q* is the outflow rate, *h* is the head of water in the storage, and *K* is the conductance of the storage outlet. Initial head in the storage is designated as *h*1.

The rate of change in the amount of water held in storage at any time is equal to the difference between inflow and outflow. Mathematically, this is expressed by the equation



 (2)

For constant *R*, the solution of equation 2 is

 (3)

where, as mentioned above, *h*1 is the head in the storage when *t* (the elapsed time) is zero. It is apparent from equation 3 that the equilibrium storage water level is given by

 (4)

where*h* in equation 4 is the water level in the storage at which recharge inflow is exactly balanced by drainage outflow, the latter being given by equation 1.

When *t* is small, equation 3 asymptotically approaches the equation

 (5)

If the initial head is zero, equation 5 becomes

 (6)

It is apparent from equations 3 to 6 that when a new recharge regime is introduced to the storage, the level of water in the storage changes to its new level in a manner that is linear at first but, with time, assymptotically approaches its final level. The time constant pertaining to the water-level adjustment process is given by the ratio of *S* to *K*. Notice that the recharge rate *R* does not figure in calculation of the time constant and that the storage coefficient *S* is not represented in the equation for the final, equilibrium water level in the storage.

Figure 6–2 shows the variation of water level in the storage with time under the assumption that the initial water level is 20 units, the recharge rate is 10−1 units, the outlet conductance is 10−3 units, and the storage coefficient is 0.2. From equation 4, it is easily established that the final equilibrium water level in the storage under these conditions is 100 units.



**Figure 6–2.** Water level in the storage depicted in figure 6–1; see text for details.

The basic storage unit pictured in figure 6–1 forms an important building block of many lumped-parameter environmental models. Hence, lessons learned in this practical session will be applicable in many circumstances where much more complex environmental models are deployed to simulate the behavior of natural systems.

## A Computer Program to Simulate the Storage

Operation of the storage depicted in figure 6–1 can be simulated by using a simple computer program based on equation 3. Such a program, written in Fortran, is listed in figure 6–3. See also file *storage.for* in the working directory for this session.

program storage

implicit none

real recharge, conductance, storage, inithead, time,

+coeff, factor

open(unit=10,file='input.dat',status='old')

open(unit=20,file='output.dat')

read(10,\*) recharge, conductance, storage

read(10,\*) inithead

read(10,\*)

coeff=(recharge/conductance-inithead)

factor=conductance/storage

write(20,10)

10 format(' Time Water\_Level')

do

read(10,\*,end=100) time

write(20,20) time,inithead+coeff\*(1.0-exp(-factor\*time))

20 format(1x,1pg14.7,2x,1pg14.7)

end do

100 end

**Figure 6–3.** A program to compute storage water levels using equation 3.

Program STORAGE reads an input file named *input.dat*. The first line of this file should contain, in order, the recharge, outlet conductance, and storage coefficient of the storage. The next line should contain the initial storage water level. A blank line should follow that, followed by a listing of the times at which water-level computation is required. A typical input file for the STORAGE program is illustrated in figure 6–4.

Figure 6–4. An input file for program STORAGE.

1.0e-1 1.0e-3 0.2 / recharge conductance storage\_coefficient

0.0 / initial head

0.1 / elapsed times

0.2

0.5

1.0

2.0

5.0

10.0

20.0

50.0

100.0

200.0

500.0

1000.0

2000.0

5000.0

10000.0

STORAGE writes an output file called *output.dat*. Figure 6–5 shows the *output.dat* file corresponding to the *input.dat* file depicted in figure 6–4.

Time Water\_Level

0.1000000 4.9987502E-02

0.2000000 9.9950016E-02

0.5000000 0.2496878

1.000000 0.4987521

2.000000 0.9950166

5.000000 2.469009

10.00000 4.877058

20.00000 9.516258

50.00000 22.11992

100.0000 39.34694

200.0000 63.21206

500.0000 91.79150

1000.000 99.32620

2000.000 99.99546

5000.000 100.0000

10000.00 100.0000

**Figure 6–5.** An output file written by program STORAGE.

## PEST++ Model Run

Theexampleproblem can be run with PEST++ by typing the command

Pest++ storage5

The output written to the screen by PEST++ and the contents of the storage5.recoutput file are presented below.

PEST++ Output Written to the Screen

C:\Users\dwelter\examples\stor>..\pest++ storage5

PEST++ Alpha Version 0.0.0

using control file: "storage5"

initializing serial run manager

OPTIMISATION ITERATION NUMBER: 1

Iteration type: base parameter solution

calculatingjacobian... (3/3 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 594.589; ending phi = 22.7623 (3.82825% starting phi)

OPTIMISATION ITERATION NUMBER: 2

Iteration type: base parameter solution

calculatingjacobian... (2/2 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 22.7623; ending phi = 14.9341 (65.6089% starting phi)

OPTIMISATION ITERATION NUMBER: 3

Iteration type: base parameter solution

calculatingjacobian... (2/2 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 14.9341; ending phi = 0.749557 (5.01909% starting phi)

OPTIMISATION ITERATION NUMBER: 4

Iteration type: base parameter solution

calculatingjacobian... (2/2 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 0.749557; ending phi = 0.438044 (58.4404% starting phi)

OPTIMISATION ITERATION NUMBER: 5

Iteration type: base parameter solution

calculatingjacobian... (2/2 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 0.438044; ending phi = 0.437872 (99.9609% starting phi)

OPTIMISATION ITERATION NUMBER: 6

Iteration type: base parameter solution

calculatingjacobian... (2/2 runs complete)

testing upgrade vectors... (7/7 runs complete)

Starting phi = 0.437872; ending phi = 0.437865 (99.9982% starting phi)

Simulation Complete - Press RETURN to close window

PEST++ storage5.rec Output File

PEST++ Alpha Version 0.0.0

Control file = storage5"

OPTIMISATION ITERATION NUMBER: 1

Iteration type: base parameter solution

Model calls so far : 0

Starting phi for this iteration Total : 594.589

Contribution to phi from observation group "OBSGROUP" : 594.589

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.727904

angle to direction of greatest descent: 45.2942 deg

Rotation Factor = 0.00 (0.00 deg); phi = 213.625 (35.93% starting phi)

Rotation Factor = 0.01 (0.41 deg); phi = 208.455 (35.06% starting phi)

Rotation Factor = 0.10 (4.19 deg); phi = 162.589 (27.34% starting phi)

Rotation Factor = 0.20 (8.59 deg); phi = 114.48 (19.25% starting phi)

Rotation Factor = 0.50 (22.65 deg); phi = 22.7623 (3.83% starting phi)

Rotation Factor = 0.70 (32.12 deg); phi = 124.385 (20.92% starting phi)

Rotation Factor = 1.00 (45.29 deg); phi = 522.574 (87.89% starting phi)

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.00184708 0.005 2.70697 0.630584

SCOEFF 0.15 0.05 3 -2

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -2.73351 -2.30103 1.18795 -0.187952

SCOEFF -0.823909 -1.30103 1.57909 0.366726

Maximum changes in transformed numeric parameters:

Maximum relative change = 0.366726 [SCOEFF]

Maximum factor change = 1.57909 [SCOEFF]

OPTIMISATION ITERATION NUMBER: 2

Iteration type: base parameter solution

Model calls so far : 10

Starting phi for this iteration Total : 22.7623

Contribution to phi from observation group "OBSGROUP" : 22.7623

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.233686

angle to direction of greatest descent: 24.3818 deg

Rotation Factor = 0.00 (0.00 deg); phi = 14.9341 (65.61% starting phi)

Rotation Factor = 0.01 (0.24 deg); phi = 15.4713 (67.97% starting phi)

Rotation Factor = 0.10 (2.39 deg); phi = 20.7701 (91.25% starting phi)

Rotation Factor = 0.20 (4.81 deg); phi = 27.5767 (121.15% starting phi)

Rotation Factor = 0.50 (12.19 deg); phi = 52.672 (231.40% starting phi)

Rotation Factor = 0.70 (17.13 deg); phi = 71.9359 (316.03% starting phi)

Rotation Factor = 1.00 (24.38 deg); phi = 101.828 (447.35% starting phi)

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.00118955 0.00184708 1.55276 0.355985

SCOEFF 0.204449 0.15 1.36299 -0.362992

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -2.92462 -2.73351 1.06991 -0.0699115

SCOEFF -0.689416 -0.823909 1.19508 0.163238

Maximum changes in transformed numeric parameters:

Maximum relative change = 0.163238 [SCOEFF]

Maximum factor change = 1.19508 [SCOEFF]

OPTIMISATION ITERATION NUMBER: 3

Iteration type: base parameter solution

Model calls so far : 19

Starting phi for this iteration Total : 14.9341

Contribution to phi from observation group "OBSGROUP" : 14.9341

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.176581

angle to direction of greatest descent: 73.4767 deg

Rotation Factor = 0.00 (0.00 deg); phi = 0.937774 (6.28% starting phi)

Rotation Factor = 0.01 (0.55 deg); phi = 0.749557 (5.02% starting phi)

Rotation Factor = 0.10 (5.90 deg); phi = 1.5011 (10.05% starting phi)

Rotation Factor = 0.20 (12.61 deg); phi = 9.06701 (60.71% starting phi)

Rotation Factor = 0.50 (36.74 deg); phi = 84.7025 (567.17% starting phi)

Rotation Factor = 0.70 (53.36 deg); phi = 147.327 (986.51% starting phi)

Rotation Factor = 1.00 (73.48 deg); phi = 179.343 (1200.90% starting phi)

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.000792229 0.00118955 1.50152 0.334008

SCOEFF 0.20644 0.204449 1.00974 -0.00973847

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -3.10115 -2.92462 1.06036 -0.0603604

SCOEFF -0.685207 -0.689416 1.00614 0.00610503

Maximum changes in transformed numeric parameters:

Maximum relative change = -0.0603604 [COND]

Maximum factor change = 1.06036 [COND]

OPTIMISATION ITERATION NUMBER: 4

Iteration type: base parameter solution

Model calls so far : 28

Starting phi for this iteration Total : 0.749557

Contribution to phi from observation group "OBSGROUP" : 0.749557

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.0343731

angle to direction of greatest descent: 79.2524 deg

Rotation Factor = 0.00 (0.00 deg); phi = 0.439216 (58.60% starting phi)

Rotation Factor = 0.01 (0.57 deg); phi = 0.438044 (58.44% starting phi)

Rotation Factor = 0.10 (6.10 deg); phi = 0.533744 (71.21% starting phi)

Rotation Factor = 0.20 (13.21 deg); phi = 0.927359 (123.72% starting phi)

Rotation Factor = 0.50 (39.63 deg); phi = 4.2616 (568.55% starting phi)

Rotation Factor = 0.70 (57.95 deg); phi = 6.85316 (914.29% starting phi)

Rotation Factor = 1.00 (79.25 deg); phi = 8.26179 (1102.22% starting phi)

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.000731946 0.000792229 1.08236 0.0760929

SCOEFF 0.206576 0.20644 1.00066 -0.000660826

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -3.13552 -3.10115 1.01108 -0.0110835

SCOEFF -0.68492 -0.685207 1.00042 0.000418703

Maximum changes in transformed numeric parameters:

Maximum relative change = -0.0110835 [COND]

Maximum factor change = 1.01108 [COND]

OPTIMISATION ITERATION NUMBER: 5

Iteration type: base parameter solution

Model calls so far : 37

Starting phi for this iteration Total : 0.438044

Contribution to phi from observation group "OBSGROUP" : 0.438044

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.00227503

angle to direction of greatest descent: 71.7733 deg

Rotation Factor = 0.00 (0.00 deg); phi = 0.437873 (99.96% starting phi)

Rotation Factor = 0.01 (0.55 deg); phi = 0.437872 (99.96% starting phi)

Rotation Factor = 0.10 (5.82 deg); phi = 0.438302 (100.06% starting phi)

Rotation Factor = 0.20 (12.42 deg); phi = 0.439937 (100.43% starting phi)

Rotation Factor = 0.50 (35.89 deg); phi = 0.453779 (103.59% starting phi)

Rotation Factor = 0.70 (52.03 deg); phi = 0.466782 (106.56% starting phi)

Rotation Factor = 1.00 (71.77 deg); phi = 0.479919 (109.56% starting phi)

Switching to central derivatives:

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.000728163 0.000731946 1.0052 0.00516841

SCOEFF 0.206736 0.206576 1.00077 -0.000771491

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -3.13777 -3.13552 1.00072 -0.000717723

SCOEFF -0.684585 -0.68492 1.00049 0.000488999

Maximum changes in transformed numeric parameters:

Maximum relative change = -0.000717723 [COND]

Maximum factor change = 1.00072 [COND]

OPTIMISATION ITERATION NUMBER: 6

Iteration type: base parameter solution

Model calls so far : 46

Starting phi for this iteration Total : 0.437872

Contribution to phi from observation group "OBSGROUP" : 0.437872

SVD information:

numver of singular values used: 2/2

upgrade vector magnitude (without limits or bounds) = 0.000495033

angle to direction of greatest descent: 79.207 deg

Rotation Factor = 0.00 (0.00 deg); phi = 0.437867 (100.00% starting phi)

Rotation Factor = 0.01 (0.57 deg); phi = 0.437865 (100.00% starting phi)

Rotation Factor = 0.10 (6.10 deg); phi = 0.437889 (100.00% starting phi)

Rotation Factor = 0.20 (13.20 deg); phi = 0.437975 (100.02% starting phi)

Rotation Factor = 0.50 (39.60 deg); phi = 0.438744 (100.20% starting phi)

Rotation Factor = 0.70 (57.92 deg); phi = 0.439424 (100.35% starting phi)

Rotation Factor = 1.00 (79.21 deg); phi = 0.439956 (100.48% starting phi)

Parameter Upgrades (Control File Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND 0.000727346 0.000728163 1.00112 0.001122

SCOEFF 0.206777 0.206736 1.0002 -0.00019962

RECHARGE 0.1 0.1 1 0

Parameter Upgrades (Transformed Numeric Parameters)

Parameter Current Previous Factor Relative

Name Value Value Change Change

------------ ------------ ------------ ------------ ------------

COND -3.13826 -3.13777 1.00016 -0.000155382

SCOEFF -0.684498 -0.684585 1.00013 0.000126624

Maximum changes in transformed numeric parameters:

Maximum relative change = -0.000155382 [COND]

Maximum factor change = 1.00016 [COND]

## PEST Model Run

The storage exampleproblem can be run using PEST by typing the command:

pest storage5

The output written to the screen by PESTand the contents of the storage5.rec output file are presented below.

PEST Output Written to the Screen

C:\Users\dwelter\Desktop\alpha\_0.0.1\stor>pest storage5

PEST Version 12.0.1.Watermark Numerical Computing.

PEST is running in parameter estimation mode.

PEST run record: case storage5

(See file storage5.rec for full details.)

Model command line:

storage1

Running model .....

Running model 1 time....

Sum of squared weighted residuals (ie phi) = 594.59

OPTIMISATION ITERATION NO. : 1

Model calls so far : 1

Starting phi for this iteration: 594.59

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 5.0000 ----->

running model .....

Phi = 178.26 ( 0.300 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 178.26

Maximum factor change: 1.747 ["cond"]

Maximum relative change: 0.7351 ["scoeff"]

OPTIMISATION ITERATION NO. : 2

Model calls so far : 4

Starting phi for this iteration: 178.26

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 2.5000 ----->

running model .....

Phi = 99.133 ( 0.556 of starting phi)

Lambda = 1.2500 ----->

running model .....

Phi = 96.104 ( 0.539 of starting phi)

Lambda = 0.62500 ----->

running model .....

Phi = 93.707 ( 0.526 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 93.707

Maximum factor change: 2.241 ["scoeff"]

Maximum relative change: 1.241 ["scoeff"]

OPTIMISATION ITERATION NO. : 3

Model calls so far : 9

Starting phi for this iteration: 93.707

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 0.31250 ----->

running model .....

Phi = 8.0233 ( 0.086 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 8.0233

Maximum factor change: 1.517 ["cond"]

Maximum relative change: 0.3410 ["cond"]

OPTIMISATION ITERATION NO. : 4

Model calls so far : 12

Starting phi for this iteration: 8.0233

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 0.15625 ----->

running model .....

Phi = 4.3394 ( 0.541 of starting phi)

Lambda = 7.81250E-02 ----->

running model .....

Phi = 4.2460 ( 0.529 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 4.2460

Maximum factor change: 1.398 ["cond"]

Maximum relative change: 0.2845 ["cond"]

OPTIMISATION ITERATION NO. : 5

Model calls so far : 16

Starting phi for this iteration: 4.2460

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 3.90625E-02 ----->

running model .....

Phi = 0.49781 ( 0.117 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 0.49781

Maximum factor change: 1.243 ["cond"]

Maximum relative change: 0.1953 ["cond"]

OPTIMISATION ITERATION NO. : 6

Model calls so far : 19

Starting phi for this iteration: 0.49781

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 1.95313E-02 ----->

running model .....

Phi = 0.43883 ( 0.882 of starting phi)

Lambda = 9.76563E-03 ----->

running model .....

Phi = 0.43913 ( 0.882 of starting phi)

Lambda = 3.90625E-02 ----->

running model .....

Phi = 0.43856 ( 0.881 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43856

Maximum factor change: 1.088 ["cond"]

Maximum relative change: 8.1179E-02 ["cond"]

OPTIMISATION ITERATION NO. : 7

Model calls so far : 24

Starting phi for this iteration: 0.43856

Calculating Jacobian matrix: running model 2 times .....

2 runs completed.

Lambda = 3.90625E-02 ----->

running model .....

Phi = 0.43788 ( 0.998 of starting phi)

Lambda = 1.95313E-02 ----->

running model .....

Phi = 0.43787 ( 0.998 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43787

Relative phi reduction between optimisation iterations less than 0.1000

Switch to central derivatives calculation

Maximum factor change: 1.010 ["cond"]

Maximum relative change: 9.4250E-03 ["cond"]

OPTIMISATION ITERATION NO. : 8

Model calls so far : 28

Starting phi for this iteration: 0.43787

Calculating Jacobian matrix: running model 4 times .....

4 runs completed.

Lambda = 9.76563E-03 ----->

running model .....

Phi = 0.43786 ( 1.000 of starting phi)

Lambda = 4.88281E-03 ----->

running model .....

Phi = 0.43787 ( 1.000 times starting phi)

Lambda = 1.95313E-02 ----->

running model .....

Phi = 0.43786 ( 1.000 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43786

Maximum factor change: 1.001 ["cond"]

Maximum relative change: 5.7673E-04 ["cond"]

Optimisation complete: the 3 lowest phi's are within a relative distance

ofeachother of 1.000E-02

Total model calls: 35

Running model one last time with best parameters.....

Recording run statistics .....

See file storage5.rec for full run details.

See file storage5.sen for parameter sensitivities.

See file storage5.seo for observation sensitivities.

See file storage5.res for residuals.

PEST storage5.rec Output File

PEST RUN RECORD: CASE storage5

PEST run mode:-

Parameter estimation mode

Case dimensions:-

Number of parameters : 3

Number of adjustable parameters : 2

Number of parameter groups : 3

Number of observations : 16

Number of prior estimates : 0

Model command line(s):-

storage1

Jacobian command line:-

na

Model interface files:-

Templates:

input.tpl

for model input files:

input.dat

(Parameter values written using single precision protocol.)

(Decimal point always included.)

Instruction files:

output.ins

for reading model output files:

output.dat

PEST-to-model message file:-

na

Derivatives calculation:-

Param Increment IncrementIncrement Forward or Multiplier Method

group type low bound central (central) (central)

recharge relative 1.0000E-02 none switch 2.000 parabolic

cond relative 1.0000E-02 none switch 2.000 parabolic

scoeff relative 1.0000E-02 none switch 2.000 parabolic

Parameter definitions:-

Name Trans- Change Initial Lower Upper

formation limit value bound bound

recharge fixed na 0.100000 nana

cond log factor 5.000000E-03 1.000000E-10 1.000000E+10

scoeff log factor 5.000000E-02 1.000000E-10 1.000000E+10

Name Group Scale Offset Model command number

rechargerecharge 1.00000 0.00000 1

condcond 1.00000 0.00000 1

scoeffscoeff 1.00000 0.00000 1

Prior information:-

No prior information supplied

Observations:-

Observation name Observation Weight Group

head1 4.998750E-02 1.000 obsgroup

head2 9.995002E-02 1.000 obsgroup

head3 0.249688 1.000 obsgroup

head4 0.498752 1.000 obsgroup

head5 0.955017 1.000 obsgroup

head6 2.66901 1.000 obsgroup

head7 4.67706 1.000 obsgroup

head8 9.81626 1.000 obsgroup

head9 21.8199 1.000 obsgroup

head10 40.8469 1.000 obsgroup

head11 63.2121 0.000 obsgroup

head12 91.7915 0.000 obsgroup

head13 99.3262 0.000 obsgroup

head14 99.9955 0.000 obsgroup

head15 100.000 0.000 obsgroup

head16 100.000 0.000 obsgroup

Control settings:-

Initial lambda : 5.0000

Lambda adjustment factor : 2.0000

Sufficient new/old phi ratio per optimisation iteration : 0.30000

Limiting relative phi reduction between lambdas : 3.00000E-02

Maximum trial lambdas per iteration : 10

Forgive model run failure during lamda testing : no

Perform Broyden's update of Jacobian matrix : no

Maximum factor parameter change (factor-limited changes) : 3.0000

Maximum relative parameter change (relative-limited changes) :na

Fraction of initial parameter values used in computing

change limit for near-zero parameters : 1.00000E-03

Allow bending of parameter upgrade vector : no

Allow parameters to stick to their bounds : no

Relative phi reduction below which to begin use of

central derivatives : 0.10000

Iteration at which to first consider derivatives switch : 1

Relative phi reduction indicating convergence : 0.10000E-01

Number of phi values required within this range : 3

Maximum number of consecutive failures to lower phi : 3

Minimal relative parameter change indicating convergence : 0.10000E-01

Number of consecutive iterations with minimal param change : 3

Maximum number of optimisation iterations : 30

Attempt automatic user intervention : no

Attempt reuse of parameter sensitivities : no

File saving options: -

Save multiple JCO files : no

Save multiple REI files : no

OPTIMISATION RECORD

INITIAL CONDITIONS:

Sum of squared weighted residuals (ie phi) = 594.59

Current parameter values

recharge 0.100000

cond 5.000000E-03

scoeff 5.000000E-02

OPTIMISATION ITERATION NO. : 1

Model calls so far : 1

Starting phi for this iteration: 594.59

Lambda = 5.0000 ----->

Phi = 178.26 ( 0.300 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 178.26

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 2.861794E-03 cond 5.000000E-03

scoeff 8.675662E-02 scoeff 5.000000E-02

Maximum factor change: 1.747 ["cond"]

Maximum relative change: 0.7351 ["scoeff"]

OPTIMISATION ITERATION NO. : 2

Model calls so far : 4

Starting phi for this iteration: 178.26

Lambda = 2.5000 ----->

Phi = 99.133 ( 0.556 of starting phi)

Lambda = 1.2500 ----->

Phi = 96.104 ( 0.539 of starting phi)

Lambda = 0.62500 ----->

Phi = 93.707 ( 0.526 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 93.707

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 2.108721E-03 cond 2.861794E-03

scoeff 0.194440 scoeff 8.675662E-02

Maximum factor change: 2.241 ["scoeff"]

Maximum relative change: 1.241 ["scoeff"]

OPTIMISATION ITERATION NO. : 3

Model calls so far : 9

Starting phi for this iteration: 93.707

Lambda = 0.31250 ----->

Phi = 8.0233 ( 0.086 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 8.0233

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 1.389632E-03 cond 2.108721E-03

scoeff 0.169866 scoeff 0.194440

Maximum factor change: 1.517 ["cond"]

Maximum relative change: 0.3410 ["cond"]

OPTIMISATION ITERATION NO. : 4

Model calls so far : 12

Starting phi for this iteration: 8.0233

Lambda = 0.15625 ----->

Phi = 4.3394 ( 0.541 of starting phi)

Lambda = 7.81250E-02 ----->

Phi = 4.2460 ( 0.529 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 4.2460

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 9.943230E-04 cond 1.389632E-03

scoeff 0.203611 scoeff 0.169866

Maximum factor change: 1.398 ["cond"]

Maximum relative change: 0.2845 ["cond"]

OPTIMISATION ITERATION NO. : 5

Model calls so far : 16

Starting phi for this iteration: 4.2460

Lambda = 3.90625E-02 ----->

Phi = 0.49781 ( 0.117 of starting phi)

No more lambdas: phi is less than 0.3000 of starting phi

Lowest phi this iteration: 0.49781

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 8.001240E-04 cond 9.943230E-04

scoeff 0.203811 scoeff 0.203611

Maximum factor change: 1.243 ["cond"]

Maximum relative change: 0.1953 ["cond"]

OPTIMISATION ITERATION NO. : 6

Model calls so far : 19

Starting phi for this iteration: 0.49781

Lambda = 1.95313E-02 ----->

Phi = 0.43883 ( 0.882 of starting phi)

Lambda = 9.76563E-03 ----->

Phi = 0.43913 ( 0.882 of starting phi)

Lambda = 3.90625E-02 ----->

Phi = 0.43856 ( 0.881 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43856

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 7.351710E-04 cond 8.001240E-04

scoeff 0.206478 scoeff 0.203811

Maximum factor change: 1.088 ["cond"]

Maximum relative change: 8.1179E-02 ["cond"]

OPTIMISATION ITERATION NO. : 7

Model calls so far : 24

Starting phi for this iteration: 0.43856

Lambda = 3.90625E-02 ----->

Phi = 0.43788 ( 0.998 of starting phi)

Lambda = 1.95313E-02 ----->

Phi = 0.43787 ( 0.998 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43787

Relative phi reduction between optimisation iterations less than 0.1000

Switch to central derivatives calculation

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 7.282420E-04 cond 7.351710E-04

scoeff 0.206731 scoeff 0.206478

Maximum factor change: 1.010 ["cond"]

Maximum relative change: 9.4250E-03 ["cond"]

OPTIMISATION ITERATION NO. : 8

Model calls so far : 28

Starting phi for this iteration: 0.43787

Lambda = 9.76563E-03 ----->

Phi = 0.43786 ( 1.000 of starting phi)

Lambda = 4.88281E-03 ----->

Phi = 0.43787 ( 1.000 times starting phi)

Lambda = 1.95313E-02 ----->

Phi = 0.43786 ( 1.000 of starting phi)

No more lambdas: relative phi reduction between lambdas less than 0.0300

Lowest phi this iteration: 0.43786

Current parameter values Previous parameter values

recharge 0.100000 recharge 0.100000

cond 7.278220E-04 cond 7.282420E-04

scoeff 0.206756 scoeff 0.206731

Maximum factor change: 1.001 ["cond"]

Maximum relative change: 5.7673E-04 ["cond"]

Optimisation complete: the 3 lowest phi's are within a relative distance

ofeachother of 1.000E-02

Total model calls: 35

The model has been run one final time using best parameters.

Thus all model input files contain best parameter values, and model

output files contain model results based on these parameters.

OPTIMISATION RESULTS

Adjustable parameters ----->

Parameter Estimated 95% percent confidence limits

value lower limit upper limit

cond 7.278220E-04 5.770578E-04 9.179754E-04

scoeff 0.206756 0.198685 0.215154

Note: confidence limits provide only an indication of parameter uncertainty.

They rely on a linearity assumption which may not extend as far in

parameter space as the confidence limits themselves - see PEST manual.

Fixed parameters ----->

Parameter Fixed value

recharge 0.100000

See file storage5.sen for parameter sensitivities.

Observations ----->

Observation Measured Calculated Residual Weight Group

valuevalue

head1 4.998750E-02 4.835774E-02 1.629764E-03 1.000 obsgroup

head2 9.995002E-02 9.669846E-02 3.251564E-03 1.000 obsgroup

head3 0.249688 0.241619 8.069500E-03 1.000 obsgroup

head4 0.498752 0.482812 1.593980E-02 1.000 obsgroup

head5 0.955017 0.963928 -8.910700E-03 1.000 obsgroup

head6 2.66901 2.39715 0.271856 1.000 obsgroup

head7 4.67706 4.75249 -7.542500E-02 1.000 obsgroup

head8 9.81626 9.34058 0.475676 1.000 obsgroup

head9 21.8199 22.1744 -0.354540 1.000 obsgroup

head10 40.8469 40.7702 7.675000E-02 1.000 obsgroup

head11 63.2121 69.4424 -6.23030 0.000 obsgroup

head12 91.7915 113.760 -21.9688 0.000 obsgroup

head13 99.3262 133.330 -34.0040 0.000 obsgroup

head14 99.9955 137.276 -37.2804 0.000 obsgroup

head15 100.000 137.396 -37.3962 0.000 obsgroup

head16 100.000 137.396 -37.3962 0.000 obsgroup

See file storage5.res for more details of residuals in graph-ready format.

See file storage5.seo for composite observation sensitivities.

Objective function ----->

Sum of squared weighted residuals (ie phi) = 0.4379

Correlation Coefficient ----->

Correlation coefficient = 0.9999

Analysis of residuals ----->

All residuals:-

Number of residuals with non-zero weight = 10

Mean value of non-zero weighted residuals = 4.1430E-02

Maximum weighted residual [observation "head8"] = 0.4757

Minimum weighted residual [observation "head9"] = -0.3545

Standard variance of weighted residuals = 5.4733E-02

Standard error of weighted residuals = 0.2340

Note: the above variance was obtained by dividing the objective

function by the number of system degrees of freedom (ie. number of

observations with non-zero weight plus number of prior information

articles with non-zero weight minus the number of adjustable parameters.)

If the degrees of freedom is negative the divisor becomes

the number of observations with non-zero weight plus the number of

prior information items with non-zero weight.

K-L information statistics ----->

AIC = -25.28434

AICC = -21.28434

BIC = -24.37658

KIC = -16.25440

Parameter covariance matrix ----->

condscoeff

cond 1.9110E-03 -3.0938E-04

scoeff -3.0938E-04 5.6236E-05

Parameter correlation coefficient matrix ----->

condscoeff

cond 1.000 -0.9437

scoeff -0.9437 1.000

Normalized eigenvectors of parameter covariance matrix ----->

Vector\_1 Vector\_2

cond 0.1603 -0.9871

scoeff 0.9871 0.1603

Eigenvalues ----->

5.9913E-06 1.9612E-03