International Ground Water Modeling Center

Prepared in cooperation with the
USGS Ground Water Resources Program (GWRP)
USGS National Water Quality Assessment Program (NAWQA)
U.S. Department of Energy

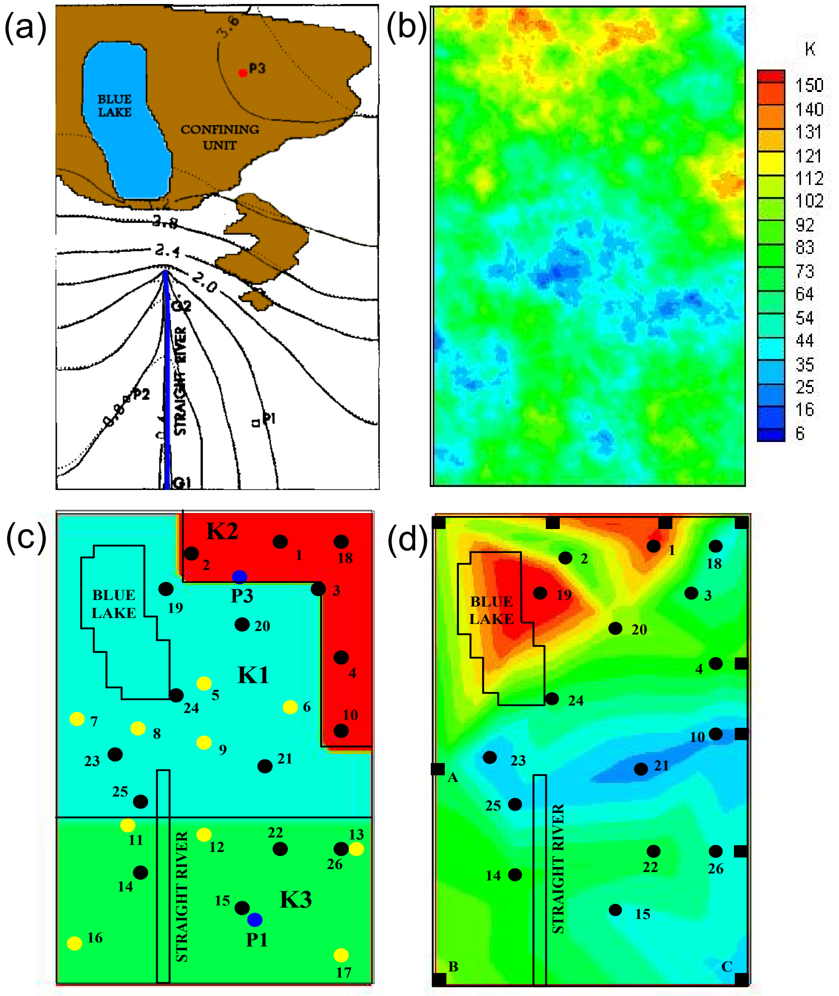
UCODE_2014, with New Capabilities to
○ Define Parameters Unique to Predictions,
○ Calculate Weights using Simulated Values,
○ Estimate Parameters with SVD,
○ Evaluate Uncertainty with MCMC,
○ and More

Constructed using the JUPITER API

JUPITER: **J**oint **U**niversal **P**arameter **I**den**T**ification and **E**valuation of **R**eliability

API: **A**pplication **P**rogramming **I**nterface

IGWMC Report number GWMI 2014-02



UCODE_2014, with New Capabilities to Define Parameters Unique to Predictions, Calculate Weights using Simulated Values, Estimate Parameters with SVD, Evaluate Uncertainty with MCMC, and More

Constructed using the JUPITER API

JUPITER: **J**oint **U**niversal **P**arameter **I**den**T**ification and **E**valuation of **R**eliability
API: **A**pplication **P**rogramming **I**nterface

By Eileen P. Poeter[1], Mary C. Hill[2], Dan Lu[3], Claire R. Tiedeman[4], and Steffen Mehl[5]

With contributions from Marc J. Rubin[6], Allan K. Haas[7], Assaf Wunsch[8]

[1] Poeter Engineering, Golden, Colorado USA. Colorado School of Mines and Integrated Groundwater Modeling Center, Emeritus. epoeter@mines.edu
[2] U.S. Geological Survey, Emeritus. Now at the University of Kansas, Lawrence, Kansas USA mchill@ku.edu
[3] Oak Ridge National Laboratory, Oak Ridge, Tennessee USA lud1@ornl.gov
[4] U.S. Geological Survey, Menlo Park, California, USA tiedeman@usgs.gov
[5] California State University at Chico, Chico, California, USA, smehl@csuchico.edu

[6] Oregon State University Cascades, School of Electrical Engineering and Computer Science, Bend, Oregon, USA marc.rubin@osucascades.edu
[7] Colorado School of Mines Department of Geophysics. Now at HydroGEOPHYSICS, Inc., Golden, Colorado, USA mygeophysics@gmail.com
[8] Leppert Associates, Golden, Colorado, USA awunsch@leppertassociates.com

For product and ordering information:
World Wide Web: http://igwmc.mines.edu

Cover figure from Hanson et al. (2013). See discussion in Chapter 6.

Suggested citation:
Poeter, Eileen P., Mary C. Hill, Dan Lu, Claire R. Tiedeman, and Steffen Mehl, 2014, UCODE_2014, with new capabilities to define parameters unique to predictions, calculate weights using simulated values, estimate parameters with SVD, evaluate uncertainty with MCMC, and more: Integrated Groundwater Modeling Center Report Number GWMI 2014-02

PREFACE

This report documents capabilities added to the computer code UCODE_2005 (Poeter et al, 2005) to produce UCODE_2014. The new capabilities include

a) support for adding parameters needed to simulate prediction conditions that were not needed to simulate calibration conditions,
b) updated weighting capabilities,
c) enhanced capabilities when using coefficients of variation to determine observations weights,
d) use of SVD parameter transformation to solve inverse problems,
e) ability to evaluate uncertainty in estimated parameters and simulated values using the DREAM Markov-chain Monte Carlo (MCMC) method, and
f) a new stacked version of a sensitivity analysis graph that allows important observations and important parameters to be identified.

Also included are instructions for using UCODE_2014 with MATLAB models and models run through a graphical interface (the latter uses HYDRUS as an example), and an example model calibration report.

No changes were needed for the post-processing programs documented with UCODE_2005. These programs are distributed with UCODE_2014, so users do not need to download items from both the UCODE_2005 and UCODE_2014 distributions of UCODE. The documentation presented in this report describes the methods used and includes detailed descriptions of the required input and typical use of the output files.

UCODE is one of a set of inverse modeling codes supported by the U.S. Geological Survey. UCODE was developed for models in which the number of parameters is less than the number of observations. It supports a wide range of parameterization capabilities through its template file and derived parameter capabilities, and emphasizes the utility of a range of sensitivity and uncertainty analysis methods. Two other inverse modeling codes supported by the U.S. Geological Survey are PEST++ (Welter et al., 2012) and bgaPEST (Fienen et al., 2013). Both support methods with which the number of parameters can exceed the number of observations. It is hoped that making a range of methods readily available to users will enable the kind of experimentation needed to understand how to enhance the predictive skill and process understanding models ideally provide. There are also a number of freely available programs from outside the U.S. Geological Survey with similar capabilities. These include, for example, PEST (Doherty 2010) and OSTRICH (Mattot, 2010). Users are encouraged to consider this rich set of alternatives as they develop their models.

The code documented by this report is public domain, open-source software and can be downloaded from the Internet at URL http://igwmc.mines.edu/. The performance of the code presented in this work has been tested in a variety of applications. Future applications, however, might reveal errors that were not detected in the test simulations. Users are requested to notify the IGWMC (Integrated Groundwater Modeling Center) of any errors found in the report or the computer codes. Updates might occasionally be made to both the report and to the codes. Users can check for updates on the Internet at URL http://igwmc.mines.edu/.

Contents

Figures

Tables

UCODE_2014, with New Capabilities to Define Parameters Unique to Predictions, Calculate Weights using Simulated Values, Estimate Parameters with SVD, Evaluate Uncertainty with MCMC, and More

Constructed using the JUPITER API

By Eileen P. Poeter[1], Mary C. Hill[2], Dan Lu[3], Claire R. Tiedeman[4], and Steffen Mehl[5]

Abstract

This report documents additions to the computer program UCODE_2005 made to create UCODE_2014. The new capabilities include support of parameters that are required for predictions but that had not been considered previously in model development. Examples of situations with such parameters are storage parameters when calibration conditions are steady-state and prediction conditions are transient, and transport parameters when calibration conditions involve only a flow field and prediction conditions include advective, advective-dispersive, or advective-dispersive-reactive transport. Second, weighting based on user-supplied coefficients of variation are supported more thoroughly in UCODE_2014 than in previous versions of UCODE and, we believe, any other available software package. Third, the use of singular value decomposition (SVD) for parameter estimation has been integrated into UCODE_2014 and the transparency offered to SVD users by sensitivity analysis is stressed. The implementation of SVD in UCODE_2014 is useful for estimating parameters in some circumstances, and guidance is provided for migrating to PEST when its unique capabilities are of interest. Inclusion of Markov-chain Monte Carlo (MCMC) uncertainty measures means that UCODE_2014 has three methods of uncertainty evaluation that

[1] Poeter Engineering, Golden, Colorado USA. Colorado School of Mines and Integrated Groundwater Modeling Center, Emeritus. epoeter@mines.edu
[2] U.S. Geological Survey, Emeritus. Now at the University of Kansas, Lawrence, Kansas USA mchill@ku.edu
[3] Oak Ridge National Laboratory, Oak Ridge, Tennessee USA lud1@ornl.gov
[4] U.S. Geological Survey, Menlo Park, California, USA tiedeman@usgs.gov
[5] California State University at Chico, Chico, California, USA, smehl@csuchico.edu

proceed from (i) linear uncertainty intervals that are very computationally frugal (commonly 10s to 100s of parallelizable model runs) and depend on the model not being too nonlinear and errors being approximately Gaussian, to (ii) nonlinear uncertainty intervals with moderate computational demands (commonly 1,000s of model runs) and depend on model smoothness and Gaussian errors, to (iii) MCMC uncertainty intervals that are computationally demanding (commonly 10,000s of model runs and more) and few restrictive assumptions. Having this range of methods in one program encourages application and exploration of the entire range of methods.

Auxiliary programs introduced since UCODE_2005 are described briefly. This includes three auxiliary programs: MMA for multi-model analysis, OPR-PPR for value of information measures, and Sim-Adjust for when simulated equivalents to observations cannot be calculated. Three graphical interfaces are also described: ModelMate for constructing and running applications of UCODE_2014, GW_Chart for evaluating model fit, sensitivity analysis results, and uncertainty measures, including support for the new capabilities, and enhancements to Model Viewer for plotting objective-function surfaces using the UCODE _sos data-exchange files.

Instructions are provided for using UCODE_2014 with programs constructed using MATLAB and models run using graphical interfaces, using HYDRUS as an example.

Example and test cases include surface-water and groundwater problems.

UCODE_2014 is constructed in a modular fashion using JUPITER API conventions and modules.

Chapter 1: INTRODUCTION

Since the publication of UCODE_2005 a great deal of activity has occurred in the field of model calibration, sensitivity analysis, and uncertainty assessment. Much of this activity has been driven by the remarkably improved computer hardware capabilities. This document describes enhancements that were made to UCODE_2005 to create UCODE_2014, and is intended to be used with the UCODE_2005 document.

Purpose and Scope

As UCODE_2005 has been used and modeling of natural and human-made systems has progressed, it has become apparent that the unique perspective provided by UCODE, which includes making each step of model development as transparent as possible, can provide an important contribution. As a result, a number of new methods have been added to UCODE to create UCODE_2014.

The main new capabilities in UCODE_2014 are enhanced support of predictions (Chapter 2), updated weighting capabilities (documented in Chapter 3 of this report), estimation of process-model parameters using singular value decomposition (SVD) parameter transformation (Chapter 4), and quantification of parameter and prediction uncertainty using Markov-chain Monte Carlo (MCMC; Chapter 5). New programs for users of UCODE and possibly other inverse models and additional modifications to UCODE are documented in Chapter 6, along with a new way of plotting composite scaled sensitivities (CSS) that clearly shows the contribution to this measure of user-defined groups of observations. Finally, Chapter 7 describes how to use UCODE_2014 with process models programmed using MATLAB, Chapter 8 shows how to use UCODE_2014 with a program run through a GUI, and Chapter 9 presents an application of UCODE to calibrate a synthetic groundwater flow model and evaluate the model predictions.

The enhanced support of predictions provides a convenient mechanism to accommodate situations in which the simulated prediction conditions require system characteristics not involved in calibration conditions. Accounting for prediction uncertainty resulting from uncertainty in prediction-specific parameters can be accomplished by defining parameters to represent those characteristics. Consider a groundwater example. Groundwater models often are calibrated using heads and flows, but are used to predict transport. If advective transport is to be simulated, the new system characteristic that needs to be considered is effective porosity. To propagate the uncertainty in effective porosity into uncertainty measures of the predicted advective transport using either Monte Carlo or inferential methods, parameters that represent effective porosity need to be defined. The new capabilities of UCODE_2014 make it possible to define such additional parameters without changing the definition of parameters already defined as part of model calibration.

The updated weighting capabilities provide more control when the user specifies coefficients of variation from which UCODE calculates observation weighting. With UCODE_2014 the user can limit the size of the weight through the use of a new input value η, even when observed values are used with the coefficients of variation to calculate the weight. With this addition, and given the ability of UCODE to use full weight matrices for observations and prior information, we believe the methods UCODE_2014 includes for weighting are among the most advanced of any open-source program of its type.

Problems with parameter insensitivity and correlation can be problematic in model calibration, and such problems can be detected using the methods described by Hill and Tiedeman (2007, p. 50-54). One way to mitigate these problems is herein-named the process-model parameter (PMP) method of obtaining a tractable regression problem. This approach identifies a set of process-model parameters to remove from the estimation process to obtain a tractable regression problem using mostly composite scaled sensitivities (CSS) and parameter correlation coefficients (PCC) (Hill and Tiedeman, 2007). All parameters with small CSS values and selected parameters with large absolute values of PCC are suggested for omission. The result is that process-model parameters are omitted that are less important because the calibration data provide minimal information about those parameters. Hill and Tiedeman (2007) suggest that the unestimated parameters need to be reactivated for uncertainty analysis.

An alternative and often more convenient way to obtain a tractable regression problem is to transform the process-model parameters into singular value decomposition (SVD) parameters, and identify a set of SVD parameters to remove from the estimation process to obtain a tractable regression. Here this is called the SVD method for obtaining a tractable regression problem. Each SVD parameter is a linear combination of process-model parameters, and the coefficients of the linear equations are calculated using SVD decomposition of weighted parameter sensitivities (the same quantities used in the PMP method to calculate CSS and PCC). SVD parameters with small singular values are suggested for omission. Small singular values tend to occur for SVD parameters dominated by process-model parameters with small CSS values and parameters with large absolute values of PCC. The SVD parameter method has become popular in part through its excellent implementation in the computer program PEST and the creative terminology “superparameters” (Tonkin and Doherty, 2005; Doherty, 2010; Doherty and Hunt, 2009, 2010a).

There are obvious parallels between the PMP and SVD parameter approaches, which are consistent with the underlying mathematics (as described in Chapter 4 and Appendix E). These parallels can be exploited to produce a sensitivity analysis that provides considerable insight about which process-model parameters are important in a model calibration problem, which observations are most important for model calibration, and which observations and parameters are important to subsequent predictions of interest. UCODE_2014 provides output to address these issues. The integration of sensitivity analysis methods from the PMP and SVD methods is, to our knowledge, unique to UCODE_2014.

An important aspect of using SVD parameters is the process of transforming SVD parameters back to process-model parameters. UCODE_2014 provides users with two methods. One method preserves the values of process-model parameters that are insensitive or extremely correlated, and one preserves the values of SVD parameters with small singular values. The latter is the method most commonly found in other works (e.g., PEST and Aster et al., 2012)

The Markov-chain Monte Carlo (MCMC) DREAM algorithm (Vrugt et al., 2009) was included in UCODE_2014 so that users would have a full suite of uncertainty evaluation tools along with a mature set of sensitivity analysis methods. UCODE_2014 can produce linear uncertainty intervals that are very computationally frugal (commonly 10s to 100s of parallelizable model runs) and depend on the model being not too nonlinear and errors being approximately Gaussian, to nonlinear uncertainty intervals with moderate computational demands (commonly 1,000s of model runs) that depend on model smoothness and Gaussian errors, to MCMC uncertainty intervals that are computationally demanding (commonly 10,000s of model runs and more) and have few restrictive assumptions. This tradeoff between increasing computational demand and decreasing restrictive assumptions and, therefore, more general applicability is important to understand when deciding what method to use in a given problem. Should one compromise the simulated processes to ensure that a computationally demanding uncertainty analysis can be used? When are the more computationally frugal methods likely to be adequate? The range of methods provided in UCODE_2014 allows appropriate methods to be used in applications and allows the different methods to be investigated by researchers using different types of problems.

Recent conceptual work in inverse modeling investigates how regression and Bayesian methods relate to one another (Lu et al., 2012). That work cites statistical literature from as far back as the 1960's that suggests conditions under which the regression method used in UCODE is a pragmatic alternative to methods for which MCMC is the modern version. The regression method in UCODE is nearly identical to the maximum a posteriori (MAP) method, though MAP is derived from a Bayesian framework. The careful evaluation of the two approaches by Lu et al. (2012) suggests that finding adequate models may often be more important than investing many runs in uncertainty intervals, but for some studies the precision of the MCMC result is vital.

Much of the attraction of Bayesian methods in the last decades has been its use of global search methods that are robust in the presence of features such as local minima typical of very nonlinear models. Recent work by Kavetski and Clark (2010) suggests that some models are far more nonlinear than the systems they represent. To the extent this is true, it may be that, if the unrealistic nonlinearities can be reduced, more computationally frugal local search methods may have greater utility than previously assumed. UCODE_2014 includes a range of methods, including gradient methods like composite scaled sensitivities (CSS), parameter correlation coefficients (PCC), and linear confidence intervals and global methods like MCMC uncertainty evaluation. Having ready access to these methods aids exploration of their relative utility.

The additional programs listed in Chapter 6 include UCODE post-processor ModelMate, new capabilities of GW_Chart, and a new Model Viewer capability to plot objective-function surfaces. Changes to keywords in several input blocks and two data-exchange files also are described. A complete updated list of data-exchange files is provided in Appendix B.

MATLAB is being used increasingly to program process models and to perform sensitivity and uncertainty analyses. It is hoped that the instructions provided in Chapter 7 for using MATLAB process-based models with UCODE will begin and evaluation of model analysis methods provided by MATLAB and UCODE.

Use of UCODE with models that are executed through a GUI is addressed using HYDRUS as an example in Chapter 8. Chapter 9 presents a groundwater flow model calibration of the synthetic Faux Valley to provide readers with an example of how model study results could be presented.

Appendix A provides a short description of the JUPITER API modules used. Appendix B lists all of the data-exchange files produced by UCODE_2014. Appendix C describes the distributed files. Appendix D describes test problems for SVD and MCMC that are distributed with UCODE_2014. Appendix E shows how the SVD ID statistic relates to the composite scaled sensitivity (CSS) reported by UCODE_2014. Appendix F presents the mathematics behind the new version of the CSS plot which uses stacked bars to show the contribution of different observation types to the estimation of the defined parameters. Appendix G provides a brief description of the capabilities of the model-independent inverse models UCODE_2014, PEST, and OSTRICH, including unique features and utility. Appendix H lists web sites with useful statistical and mathematical capabilities such as normal probability calculations. Appendix I describes how to run UCODE_2014 and provides abbreviated input instructions for all UCODE_2014 capabilities, including those documented in the UCODE_2005 report.

A Word about Words: Process Model

Both the UCODE_2005 documentation and this UCODE_2014 report use the phrase “process model” to describe the model run by UCODE and for which UCODE estimates parameter values. Consider this within the ongoing debate in many fields of environmental science about three types of models: statistical models based on probability distribution functions (pdfs), conceptual models based largely on if/then rules, and process-based models based largely on differential equations. The authors of UCODE have used process-based models throughout their careers, and when defining a term by which to differentiate the model run by UCODE and the inverse model embodied by UCODE, it was natural to call the model run by UCODE a process model. However, UCODE is not limited in any way, and is equally capable of being used with statistical and conceptual models (for example, Foglia et al., 2013 evaluate a range of conceptual models that differ in how aspects of a system are simulated).

Thus, UCODE is not limited to process models as defined in some scientific disciplines, despite the use of the term “process model”. Of concern is whether the parameters to be estimated and the observations used in the estimation are real numbers (UCODE applies) or discrete (UCODE does not apply). In some cases with discrete parameter values or for other aspects of model construction for which discrete alternatives are of interest (for example, alternative ways of calculating recharge or alternative conceptualizations of boundary conditions or geometry of system characteristics), constructing alternative models that represent the discrete alternatives and using UCODE and another code called Multi-Model Analysis (MMA) (Poeter and Hill, 2007) can be useful. For an example, see Foglia et al. (2013).

Acknowledgements

This Faux Valley test case described in Chapter 9 was originally developed to teach a short course in February, 2009. The authors wish to thank Eve Kuniansky for challenging us to create a simple “report” that will hopefully be useful to students learning how inverse modeling and model analysis methods can be used.

The authors gratefully acknowledge reviews by Dr. Laura Foglia, Technical University Darmstadt, Dr. Stefan Finsterle at Lawrence-Berkeley National Laboratory, and Professor David Tartakovsky at the University of California, San Diego. We appreciate their many comments and incisive, constructive questions.

.

Chapter 2: PREDICTION RUNS WITH NEW PARAMETERS

Simulation models of natural and engineered processes commonly are developed and calibrated for the purpose of prediction. Prediction simulations generally differ from the calibration simulations. Examples related to groundwater systems include the following.

(1) The nature of the prediction may differ from the observations, as occurs when the model is calibrated using heads and flows and predictions include the position of particles tracked in the flow field or simulated values of concentration. For either type of transport prediction, parameters defining the distribution of effective porosity are needed for the predictive model but not for the calibration model. For concentrations, generally parameters characterizing dispersive processes also are needed.
(2) The model may be calibrated for time-averaged steady-state conditions, and the predictions are inherently transient, such as those needed to evaluate proposed aquifer storage and recovery (ASR) scenarios. In this case, parameters defining storage properties are needed for the predictive model but not for the calibration model.
(3) Part of the system may be removed or altered due to mining activity or natural dissolutions during karstification. For this case, changes in boundary conditions and the addition of turbulent flow equations may be needed for the predictive model but not for the calibration model.

The new parameters are important in part because any uncertainty in their assigned value produces uncertainty in the predictions. Prediction uncertainty can be approximated using UCODE_2014 and related programs in four ways:

(a) Linear uncertainty intervals calculated by the UCODE post-processor LINEAR_UNCERTAINTY (documented by Poeter et al., 2005).
(b) Nonlinear uncertainty intervals calculated by UCODE.
(c) The new MCMC capability described in Chapter 5 of this work.
(d) Model averaged uncertainty intervals calculated by the program MMA (Poeter and Hill, 2007).

In calculating measures of prediction uncertainty, it is important to include any knowledge about the new parameters in prior information equations. Omission of the defensible prior information on these parameters can lead to larger measures of parameter uncertainty. If correlation between the added parameters and the calibration parameters is of importance, the new parameters need to be included in the originally defined input blocks and the convenience of the new input blocks provided by UCODE_2014 cannot be used. Of note is that when aspects of a model important to predictions are not informed by observations, prediction accuracy often suffers. This point is made by Oreskes et al. (1994) and is demonstrated vividly by Doherty and Christensen (2011).

In addition to measures of prediction uncertainty, the following are also of interest:

(i) Prediction scaled sensitivities (PSS) calculated by UCODE and Parameter-Prediction (PPR) statistics calculated by OPR-PPR (Tonkin et al., 2007) can be used to identify parameters important to predictions.

(ii) Observation-PRediction (OPR) statistics calculated by the program OPR-PPR can be used to identify potential new observations likely to reduce prediction uncertainty, and to identify the observations used in model calibration that are contributing to very small prediction uncertainty measures.

While the evaluations needed when there are parameters that apply only to prediction conditions could be done with UCODE-2005, these evaluations are more easily conducted using the new capabilities in UCODE_2014.

Without the new capabilities described in this chapter, the following procedure is needed. The basic issue is the parameters defined as active in a sensitivity or parameter-estimation run that simulates calibration conditions need to be identical to those in the prediction run (Table 2 of the UCODE_2005 documentation describes these runs). The first calculates the a posteriori matrix of parameter variances and covariances for the calibration conditions; the second calculates a vector of sensitivities for the predictive conditions. The dimensions of the matrix, which is square, need to match the dimensions of the vector. This is accomplished by the runs described. The prediction-only parameters need to be included in the first run even though the observation sensitivities to these parameters equal zero. An example is described in Hill and Tiedeman (2007, Chapter 8 exercises). Defining these prediction-only parameters in the sensitivity or calibration run can be cumbersome and prone to error.

Any parameters added for predictions using the capabilities documented here need to have zero sensitivity relative to the observations used to estimate parameters. If there is any question, calculate the sensitivities related to observations using the parameters in question and check that the sensitivities are zero. This can be done by checking that dimensionless and composite-scaled sensitivities (DSS and CSS) are zero. Values of zero for CSS identify parameters that are completely unimportant to the predictions. In general it is best to look for zeros instead of just small numbers. Though numerical issues can produce small non-zero values that are acceptable, they may not be conclusive. This is especially true for parameters with a datum (see discussion in Hill and Tiedeman, 2007, p. 57).

This chapter describes the UCODE_2014 capability of defining parameters and prior information only for prediction runs, and describes the associated UCODE model output.

The capabilities described here were originally included in UCODE_2005 version 1.009. At that time it was documented using an auxiliary pdf file distributed with UCODE_2005. This section replaces that file. The capability remains the same and is fully backward compatible.

New "_For_Prediction" Input Blocks

Six input blocks accommodate the UCODE_2014 support of parameters that are only applicable to prediction conditions. The blocklabels for the new input blocks are shown in bold type in Table 1, in the order in which the new input blocks need to appear in the main UCODE_2014 input file relative to other input blocks. The name of each new input block builds on the name of an existing input block, by appending "_For_Prediction" to the name of the existing block, as shown in Table 1.

For each of the "For_Prediction" input blocks, the keywords are exactly the same as those for the existing input block that has the same name without "_For_Prediction" appended. These inputs are documented in Poeter et al. (2005). Abbreviated input instructions are provided in Appendix I of this report.

Parameter_Groups_For_Prediction (optional)

Use this input block to assign data that apply to all or many of the parameters defined only for the predictions. When quantities specified in the Parameter_Groups_For_Prediction input block are repeated in the Parameter_Data_For_Prediction block, the data specified in the Parameter_Data_For_Prediction block are used.

Keywords available for use in this input block are the same as those available for the Parameter_Groups input block described fully in Poeter et al. (2005, p. 68). The keywords and brief descriptions are listed in Appendix I of this report.

Parameter_Data_For_Prediction (optional)

Use this input block to assign data to parameters defined only for the predictions and for which uncertainty in the parameter is to be propagated into the prediction uncertainty. If any knowledge about added parameters exists, use it to define prior information in the Linear_Prior_Information_For_Prediction input block. Instructions for the Parameter_Data_For_Prediction input block are identical to those for the UCODE_2005 Parameter_Data input block, as documented in Poeter et al. (2005, p. 69-72). The keywords are listed in Appendix I of this report.

Parameter_Values_For_Prediction (optional)

Use this input block to specify the values of parameters defined in the Parameter_Data_For_Prediction input block. Values listed in this input block replace values listed under keyword StartValue of the Parameter_Data_For_Prediction input block. Keywords in this input block are described in Poeter et al. (2005, p. 74) and are listed in Appendix I of this report.

Derived_Parameters_For_Prediction (optional)

Use this input block to define parameters using other parameters. The other parameters are defined in the Parameter_Data_For_Prediction input block or in preceding lines of the Derived_Parameters_For_Prediction input block. Keywords in this input block are described in Poeter et al. (2005, p. 76) and are listed in Appendix I of this report.

Prior_Information_Groups_For_Prediction (optional)

Use this input block to assign data that apply to many or all of the items of prior information defined in the Linear_Prior_Information_For_Prediction input block. When quantities specified in the Prior_Information_Groups_For_Prediction input block are repeated in the Linear_Prior_Information_For_Prediction block, the data specified in the Linear_Prior_Information_For_Prediction block are used.

Keywords in this input block are described in Poeter et al. (2005, p. 94) and are listed in Appendix I of this report.

In some circumstances, the prior information for the parameters used only by predictions includes parameter correlation. This can be represented by defining the relevant variance-covariance matrix in the Matrix_Files input block and listing the name of that matrix in the Prior_Information_Groups_For_Prediction input block using the CovMatrix keyword.

Linear_Prior_Information_For_Prediction (required by post-processor LINEAR_UNCERTAINTY when Parameter_Data_For_Predictions input block exists)

Use this input block to assign prior information on parameters defined only for prediction conditions. The instructions for this input block are the same as those for the Linear_Prior_Information input block described in Poeter et al. (2005, p. 95-96). The keywords and brief descriptions are listed in Appendix I of this report.

Expressions defined for the Equation keyword need to consist only of the names of parameters listed in the PARAMETER_DATA_FOR_PREDICTION input block. To define a prediction-only parameter using a parameter listed in the Parameter_Data input block, add the prediction-only parameter to the Parameter_Data or the Derived_Parameters input block. That is, while the _For_Prediction input blocks can be a convenient way to manage parameters needed only for prediction conditions, an alternative is to add them to the original input blocks and this is required if the prediction-only parameters are to be expressed in terms of the other parameters.

The Linear_Prior_Information_For_Prediction input block is required if the Parameter_Data_For_Prediction input block exists and UCODE post-processors LINEAR_UNCERTAINTY (Poeter et al., 2005) and(or) OPR-PPR (Tonkin et al., 2007) are used. Because these parameters are not defined for the calibrated model, prior information is the only mechanism for including any knowledge about these parameters in measures of prediction uncertainty. The definition of prior information on these parameters is accomplished using the Linear_Prior_Information_For_Prediction input block and, optionally, the Prior_Information_Groups_For_Prediction input block. If there is no information about an added parameter, the lack of knowledge can be expressed by defining prior information and assigning it a large value of variance, standard deviation, or coefficient of variation, or a small weight. These are assigned using the keywords Statistic and StatFlag.

Output Changes

Changes to UCODE output are mostly associated with existing and new data-exchange files, which are described here. Changes to the main output file depend in the value of Verbose in the Options input block and are clearly labeled.

When there are parameters defined only for predictions, there are changes to seven existing data_exchange files and four new data_exchange files are created. None of the changes are to files commonly accessed by the user. These files are primarily used by other programs such as Linear_Uncertainty, OPR-PPR, and MMA.

Changes to Existing Data-Exchange Files

Data-exchange file ***_dmp*** includes a second line with the label "NUMBER OF PARAMETERS FOR PREDICTIVE EVALUATION = ", which reports the total number of parameters defined for prediction conditions.

Data-exchange files ***_sppp***, ***_sppr***, ***_spsp***, ***_spsr***, and ***_spu*** include sensitivities for parameters defined only for prediction conditions.

Data-exchange file ***_pv*** has the same appearance, but calculated prediction variances include the effects of prediction sensitivities for parameters defined only for prediction conditions, and of the uncertainty in these parameters. The effect of any prior information on these parameters is included in the reported measures of prediction uncertainty.

New Output Files

The four new data-exchange files have filename extensions ***_mvp, _paoptp, _suprip*** and ***_wtprip***. Each is similar to an existing data-exchange file for which the filename extension lacks the final "p".

File ***_mvp*** has contents similar to the file ***_mv*** (Poeter et al., 2005, Table 19, p. 154), except only the parameters for prediction are listed.

File ***_ paoptp*** has the same contents as file ***_ paopt*** (Poeter et al., 2005, Table 19, p. 154), except only the parameters for prediction are listed.

File ***_suprip*** has the same contents as file ***_supri*** (Poeter et al., 2005, Table 18, p. 153), except PRIOR NAME is the name of a prior information equation defined in the Linear_Prior_Information_For_Prediction input block, and Param-Name1, Param-Name2, and so on, are names of parameters defined in the Parameter_Data_For_Prediction input block.

File ***_wtprip*** has the same contents as file ***_wtpri*** (Poeter et al., 2005, Table 17, p. 152), except the number of rows equals the number of prior information equations defined in the Linear_Prior_Information_For_Prediction input block.

Example UCODE_2014 Input

In this example, a transient model is calibrated using head, drawdown, and flow observations, and is then used to predict an advective transport path. The prediction requires that an effective porosity parameter be defined; this parameter is not applicable to the calibration run. The part of the UCODE_2014 input file for the prediction run shown in Figure 1 lists input blocks that pertain to the definition of parameters, predictions, and prior information.

```
# ---------------------
# PARAMETER INFORMATION
# ---------------------
BEGIN PARAMETER_GROUPS KEYWORDS
   GroupName = MyPars  adjustable=yes TOLPAR=.01 SenMethod=2
   MaxChange=2.0
END PARAMETER_GROUPS

BEGIN PARAMETER_GROUPS_FOR_PREDICTION KEYWORDS
   GroupName=porgroup  Adjustable=yes TOLPAR=0.01    SenMethod=2
      MaxChange=2.0    Transform=no   Scalepval=0.33
END PARAMETER_GROUPS_FOR_PREDICTION

BEGIN PARAMETER_DATA FILES
..\ex1b-files\tr.params
END PARAMETER_DATA

BEGIN PARAMETER_DATA_FOR_PREDICTION TABLE
nrow=1  ncol=9 columnlabels
paramname STARTVALUE lowervalue uppervalue perturbamt groupname
  POR_1&2   0.33        0.27       0.39     0.01D0    porgroup
END PARAMETER_DATA_FOR_PREDICTION

# ----------------------------
# PREDICTION INFORMATION
# ----------------------------
BEGIN PREDICTION_GROUPS FILES
..\ex1b-files\groups.pred
END PREDICTION_GROUPS

BEGIN PREDICTION_DATA FILES
..\ex1b-files\adv.pred
END PREDICTION_DATA

...

# ----------------------
# PRIOR INFORMATION
# ----------------------
BEGIN PRIOR_INFORMATION_GROUPS_FOR_PREDICTION KEYWORDS
GroupName=prior   PlotSymbol=4  UseFlag=yes
END PRIOR_INFORMATION_GROUPS_FOR_PREDICTION

BEGIN LINEAR_PRIOR_INFORMATION_FOR_PREDICTION KEYWORDS
PriorName=PrPorosity     Equation=POR_1&2   PriorInfoValue=0.33
Statistic=0.03           StatFlag=SD
GroupName=prior
END LINEAR_PRIOR_INFORMATION_FOR_PREDICTION
```

Figure 1. Part of UCODE_2014 main input file ***15.in*** showing input blocks that define an effective porosity parameter that is only applicable to the prediction run, and of prior information on this parameter. Input blocks described in this document are shown in bold type.

Chapter 3: ENHANCED OBSERVATION WEIGHTING

UCODE_2005 supports the use of error-based weighting in model development. Error-based weighting has appeared in the literature extensively (for example, Hill, 1992; Hill, 1998; Mroczkowski et al., 1997; Oliver et al., 2008; Foglia et al., 2009, Renard et al., 2011), and is the only objective formulation that in ideal circumstances produces minimum variance parameter estimates (Hill and Tiedeman, 2007, p. 396-398). For error-based weighting in UCODE_2005, the user has been able define a full variance-covariance matrix to characterize observation errors that are dependent, or independent errors can be characterized using values specified for each observation. The enhancements introduced in UCODE_2014 involve only the situation when individual values for each observation are specified.

To support error-based weighting when observations errors are independent, UCODE has always allowed the user to specify the statistic most applicable to a given situation, and has calculated the needed weight using that information. The statistics that can be specified include a coefficient of variation (the standard deviation of the observation error divided ideally by the true value of the observation), a standard deviation, and a variance. UCODE_2005 also allows the weight and square-root of the weight (the latter provided for compatibility with PEST). Of these alternatives, the enhancements provided by UCODE_2014 only involve how coefficients of variation can be used to calculate weights.

Coefficients of variation are generally used when the observation errors are thought of in terms of percent or fraction of the observed value. This most often happens when

1. The quantity involved can vary over many orders of magnitude, as occurs for concentrations and streamflow. For example, errors in streamflow are often thought of as about 5 to 10 percent of the measured flow if measurement error dominates, or larger if epistemic model error is thought to be significant.
2. An arbitrary datum is not needed to define the value. An arbitrary datum is used, for example, for groundwater heads defined relative to sea level. Sea level is an arbitrary datum; it could just as well be the elevation of some point within the area of interest. For example, consider a system in which heads are 1,000 meters (m) above sea level. For this system, changing the datum to 900 m above sea level results in the observed heads being around 100 m. The same flows and relative heads would be simulated – just the datum has changed. This is what we call an arbitrary datum – the modeler can choose any datum. Coefficients of variation are generally not valid for defining weights for observations for which the datum can be defined arbitrarily.

In the UCODE_2005 documentation, conversion of coefficients of variation to weights is defined in equation 3. The equation is

$$\omega_{i_i} = \frac{1}{\left(cv_i \tilde{y}_i + \eta\right)^2} \qquad (1)$$

where i identifies the observation, ω_i is the weight, cv_i is the coefficient of variation, $\tilde{y}_i$ is the true value, and η is a constant that can be added to control how much small values of y_i influence the results.

As noted there, the true value is not known. Obvious possibilities for its estimation include the observed and simulated value. The results of Anderman and Hill (1999) suggest that under ideal conditions slightly less parameter bias can be obtained using simulated values. However, using simulated values from the beginning of the regression can produce weights that vary dramatically from one parameter-estimation iteration to the next. Thus, even when the goal is to use simulated values, it is generally advantageous to start out using the observed values. UCODE_2014 provides new alternatives for using coefficients of variation.

Change in How Coefficients of Variation are used to Calculate Weights

In UCODE_2014, an observation value for which a coefficient of variation is used to define the weighting can either (1) use the observed value in equation 1 or (2) use the observed value at first and use the simulated value as regression approaches a solution. For both, the value of η is specified by the user and applied to calculate the weight for all parameter estimation iterations. This differs from UCODE_2005 in which use of the observed value only (option 1) allowed only η=0.0.

Another change in UCODE_2014 captures a potential user-input error. If the denominator of equation 1 is calculated as zero, a zero is assigned to the weight of that observation. This will occur if $\tilde{y}_i$ and η are both zero. The code tests for zero values of cv_i (the user-defined input) when UCODE begins; if cv_i =0.0, an error message is printed and execution stops.

The following section discusses how to use η and how weighting with coefficients of variation is controlled in UCODE_2014.

Using η

Figure 2 shows the influence of η on the value of the square-root of the weight. A wide range of $\tilde{y}_i$ values (labeled as y in the figure) is shown because observations may not be normalized. Of interest is what values to use for η.

When η = 0.0, Figure 2 shows that the value of the weight can become very large for small values of y. This may make sense. For example, streamflows may be expected to have an accuracy of 10% over a wide range.

For other problems such large weights for small values may be problematic. For example, again consider streamflow. If groundwater processes are not included in the simulation and are important, simulated low flows are likely to be less accurate than high flows (actually biased because they lack an important source of water). In this case the larger weights produced for low flow when η = 0.0 than for η > 0.0 would likely be ill-advised. In such circumstances η > 0.0 can be useful.

What value of η should be used? Commonly the value of η is set equal to or slightly larger than a typical residual value given known error contributions (measurement and model). The value of η can also be determined by what value is significant to a given problem. For example, if concentration values are normalized so that the largest value equals 1.0, and values less than 1×10^{-4} are considered to be small enough to lack significance for the problem considered, a value of $\eta = 1 \times 10^{-4}$ could be appropriate.

Data range issues can also be accommodated using other capabilities in UCODE. For example, the detection limit capability described for the UCODE_2005 documentation on page 84 using keyword Nondetect may in some circumstances be useful. For more information, see the Observation_data input block instructions in the UCODE_2005 documentation.

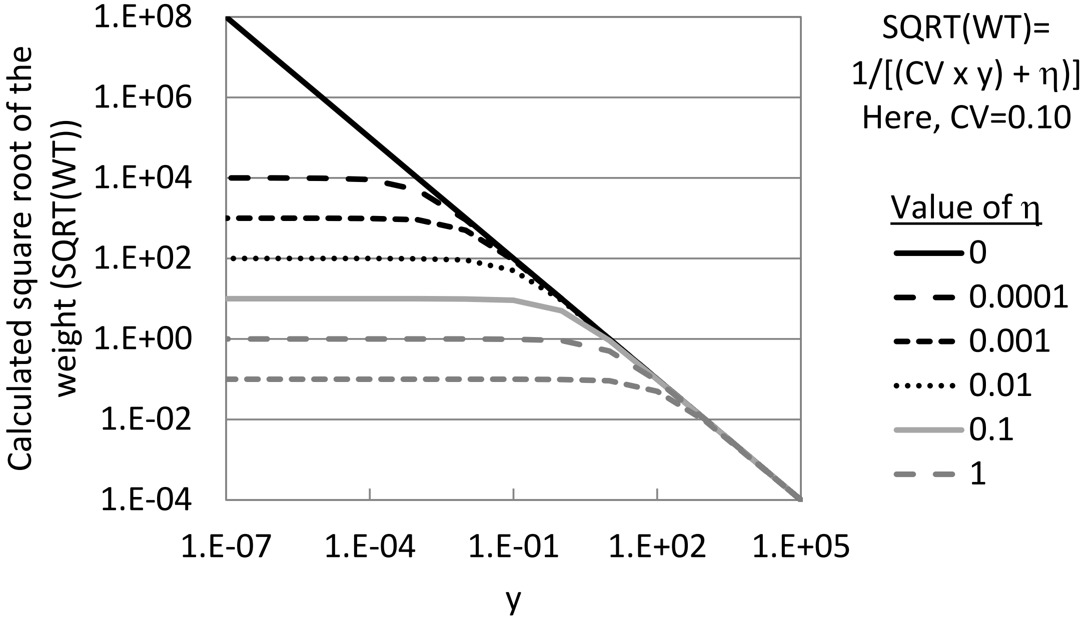


Figure 2. Weights calculated with η equal to six values between 0.0, 1.0. The true value $\tilde{y}_i$ (labeled y in the figure) is generally approximated using either the observed or simulated value. In either case, this figure shows the effect of η in equation (1). SQRT, square root; WT, weight; CV, coefficient of variation.

Modified Input

Keywords that can be used in three input blocks are modified: UCODE_Control_Data, Reg_GN_Control, Observations_Groups, and Observation_Data. The keyword changes are listed first and then their utility is described.

UCODE_Control_Data Input Block

Keyword Use_WT is added to enable UCODE runs that follow regression runs to use the final weights from the regression. These will differ from weights calculated from the Observation_Data input block when weights are calculated from coefficients of variation using simulated values, as discussed in this chapter.

Use_wt yes/no. Yes uses the weights saved in the _wt data-exchange file to calculate the likelihood; no uses the weights calculated based on the statistics defined in the Observation_Data input block. **Default=no**.

When coefficients of variation (CV) are used to define weights in the Observation_Data input block, the final weights saved in the _wt file may be different from the weights calculated based on the Observation_Data input block information, as described in Chapter 3 of this documentation. The issue occurs because simulated values instead of observed values may be used to calculate the weights. To use the _wt file produced by a previous UCODE regression, set Use_wt to yes. If Use_wt=yes and a _wt file does not exist, an error message is printed and execution stops. When statistics other than CV are used to calculate weights, or if observed values are used to convert the CV values to weights, the weights in the _wt file are the same with those calculated based on the observation_data input block. If this applies to all observations, Use_wt = no can be used by designation or default.

Reg_GN_Control Input Block

Keyword TolParWtOS functions as for UCODE_2005. It is included here for completeness and is described more thoroughly in the UCODE_2005 manual. Keyword TolParWtOSRenew is new.

The basic situation is shown in Figure 3. The keyword definitions follow.

If simulated values are used at the beginning of the parameter estimation when parameter values typically are changing a lot, the simulated values will change a lot and, if used to determine the weights, produce an unstable estimation process. Using the observed values until the parameter changes are smaller helps ensure the estimation process proceeds more smoothly. As regression approaches convergence, even small changes in the simulated values used to determine the weights could impede convergence despite the changes in simulated values being small. Thus, as convergence is reached, as evidenced by the change in parameters becoming small the simulated value used to determine the weight is no longer updated.

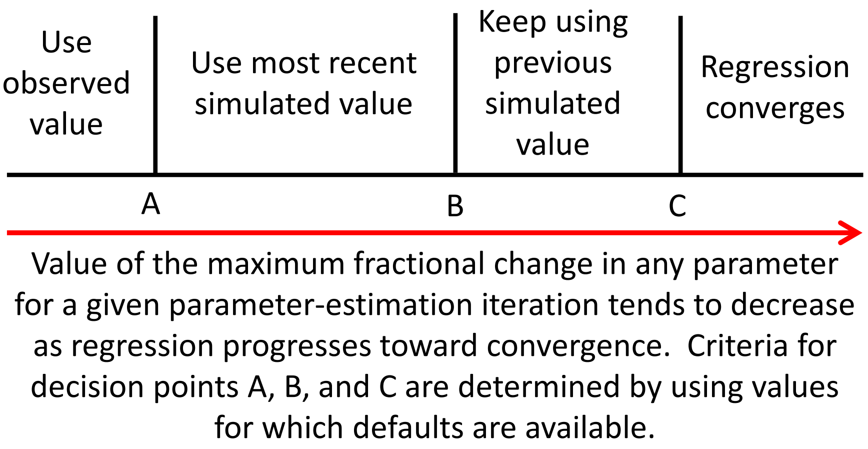

Use observed value
Use most recent simulated value
Keep using previous simulated value
Regression converges
A
B
C
Value of the maximum fractional change in any parameter for a given parameter-estimation iteration tends to decrease as regression progresses toward convergence. Criteria for decision points A, B, and C are determined by using values for which defaults are available.

Figure 3. Conditions for using observed and simulated values to calculate weights when $\tilde{y}_i$ of equation 1 can be a simulated value.

The new keywords for the Reg_GN_Control input block are as follows.

TolParWtOS a value used to calculate A and B of Figure 3 as

A = TolParWtOS×TolPar and

B = (TolPar + 0.1×(TolPar × TolParWtOS – TolPar)).

Generally larger than 5. **Default=10**.

TolParWtOSRenew Options: true, false. Determines what happens after the first time the maximum calculated parameter change is less than criterion B of Figure 3. **Default=False**.

True: Weights based on simulated values are calculated using the following rules (A, B, and C refer to criteria listed in Figure 3).

(1) They are recalculated using observed values after an iteration in which any parameter change is above the upper threshold A = TolParWtOS×TolPar,

(2) Subsequently they are recalculated using simulated values whenever all parameter changes fall between the two thresholds {i.e., below [A = TolParWtOS×TolPar] and above [B = TolPar + 0.1×(TolPar × TolParWtOS – TolPar)]}, and

(3) They are held constant whenever all parameter fractional changes are below the threshold [B = TolPar + 0.1*(TolPar * TolParWtOS – TolPar)] and any are above threshold C (TolPar).

False: Once an iteration is reached in which all of the calculated parameter changes have fallen below (B = TolPar + 0.1*(TolPar * TolParWtOS – TolPar)), then the weights are held constant for all future iterations, even when subsequent iterations have maximum parameter value changes that exceed criterion B.

Observation_Groups and Observation_Data Input Blocks

Keyword WtOSUse is new; WtOSConstant is included for completeness.

WtOSUse **-** OBS, ConvertOS, Sim, or None.

OBS, use observed value in equation 1.

ConvertOS, start using observed value and convert to the simulated value as regression approaches convergence. Performance is governed by TolParWtOS and TolParWtOSRenew of the Reg_GN_Controls input block

SIM, use simulated values in equation 1

NONE, use observed value with η=0.0 in equation 1.

Default: if WtOSConstant=0.0, default is OBS; if WtOSConstant>0.0, default is ConvertOS.

WtOSConstant the value of η used in equation 1. Default is $\eta = 0.0$

Using the Modified Input

User input that controls the weighting involves keywords WtOSConstant and TolParWtOS, which were defined for UCODE_2005. Use of these two keywords is backward compatible. One new keyword is defined: WtOSUse. The letter sequence “WtOS” used in these keywords stands for “Weights calculated using coefficients of variations and Observed and(or) Simulated values”. The keywords perform as follows.

If an observation is assigned WtOSConstant > 0 and StatFlag does not equal CV the program terminates with the message

“WEIGHTING BASED ON OBS&SIM VALUES REQUIRES StatFlag = CV”.

UCODE_2014 is backward compatible as follows. If an observation is assigned StatFlag=CV, and the new variable WtOSUse is set to ConvertOS by designation or default, then UCODE_2014 proceeds to use weights based on observed and simulated values as described for UCODE_2005.

If StatFlag=CV, WtOSUse=ConvertOS, and TolParWtOSRenew=False, UCODE_2014 performs as shown in Figure 4.

For example, if TolPar is set to the default of 0.01 for every parameter and TolParWtOS is the default 10, then the following sequence of values is used.

- at the start of the regression, weights are based on the observed value.
- after the calculated change for all parameters is less than 0.10, weights are based on new simulated values at each iteration.

- after the calculated change for all parameters is less than 0.019, weights are no longer updated with new simulated values for subsequent parameter iterations. This allows for stable conditions as the regression attempts to reach convergence.

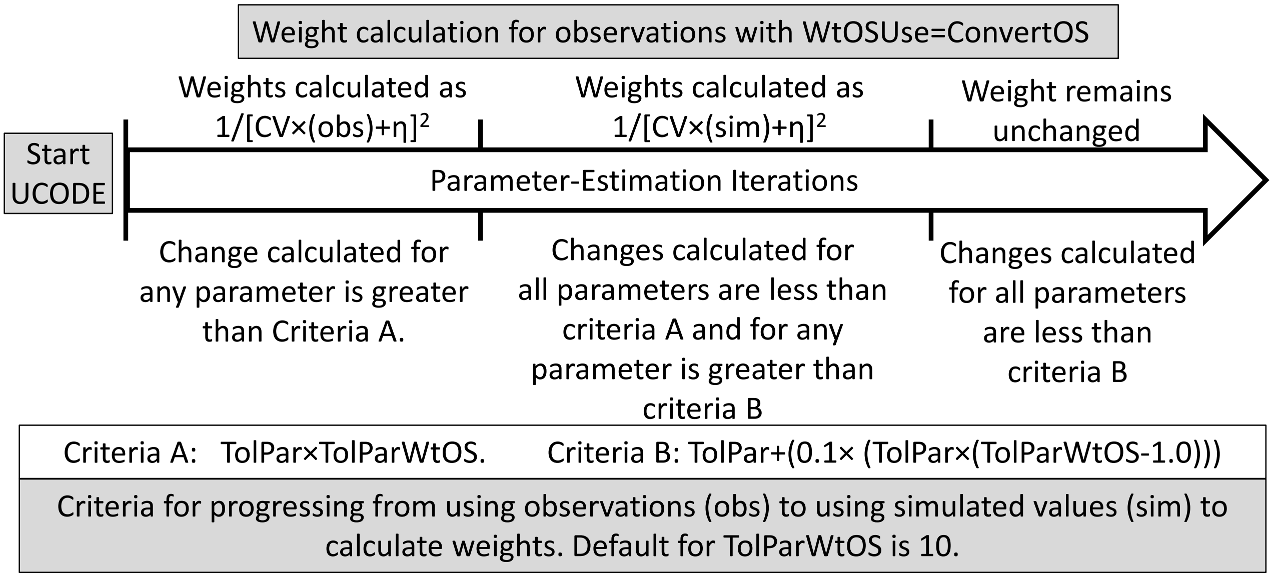

Weight calculation for observations with WtOSUse=ConvertOS
Start UCODE
Weights calculated as $1/[CV\times(obs)+\eta]^2$
Weights calculated as $1/[CV\times(sim)+\eta]^2$
Weight remains unchanged
Parameter-Estimation Iterations
Change calculated for any parameter is greater than Criteria A.
Changes calculated for all parameters are less than criteria A and for any parameter is greater than criteria B
Changes calculated for all parameters are less than criteria B
Criteria A: TolPar×TolParWtOS.
Criteria B: TolPar+(0.1× (TolPar×(TolParWtOS-1.0)))
Criteria for progressing from using observations (obs) to using simulated values (sim) to calculate weights. Default for TolParWtOS is 10.

Figure 4. Calculation of weights when TolParWtOS>0, WtOSUse=ConvertOS, and TolParWtOSRenew=False. The criteria listed are used as shown in the figure to control whether weights are calculated using observations (obs) or simulated values (sim). Figure 3 shows the concept; this figure shows how the values of the keywords fit into the process.

If an observation is assigned StatFlag=CV and the new variable WtOSUse is assigned as “OBS”, then UCODE_2014 uses one weight throughout the regression calculated with the observed value for the true value in equation 1 and η=WtOSConstant.

If an observation is assigned StatFlag=CV, and the new variable WtOSUse is assigned as “SIM” then UCODE_2014 uses a different weight for every parameter-estimation iteration calculated with the simulated value for the true value in equation 1 and η=WtOSConstant.

If an observation is assigned StatFlag=CV, and the new variable WtOSUse is assigned “NONE” then UCODE_2014 uses one weight throughout the regression calculated with the observed value for the true value in equation 1 and η=0.0.

The weight calculations described above are controlled using keywords WtOSConstant and WtOSUse in the Observation_Data input blocks. When a conversion between using observed and simulated values to calculate weights is indicated, TolParWtOS of the Reg_GN_Controls input block is used to determine when during the regression the conversion takes place.

Once regression converges, the final weight is considered the best reflection of observation error, and is used in all further calculations.

Chapter 4: SOLVING INVERSE PROBLEMS USING SVD

Singular-value decomposition (SVD) provides a more convenient way to obtain a tractable regression problem when, as is common, parameter insensitivity and dependence make estimating all defined parameters infeasible. Compared to the process of identifying insensitive and dependent parameters using CSS and PCC and deciding what parameters to omit in an ad hoc process such as that described by Hill and Tiedeman (2007), SVD addresses these problems in a more robust manner. In addition, our limited unpublished testing and the results of Hill and Østerby (2004) suggest that the estimation space can have slightly higher dimension when it is defined using SVD parameters. That is, slightly more parameters can be estimated with SVD than using other methods.

A disadvantage of SVD is that it can be a black box that obscures insights about important and unimportant process-model parameters and observations obtainable from the calculated sensitivities. Such insights are needed to understand consequences on model performance of choices made in model construction, detect model conceptualization and input errors, and guide data collection and model enhancement. Important insights might include, for example, identifying model parameters and observations that dominate simulated values of interest and contributors to calculated measures of uncertainty. Foglia et al. (2013) illustrate how what may seem like inconsequential decisions in model development can make large differences in predictions and measures of prediction uncertainty.

The UCODE_2014 documetation and program are designed to make such insights as accessible as possible to the user. First, the background of SVD is presented in a way that we believe is clearer to non-mathemeticians than most presentations. While understanding of the rather complicated mathematics behind SVD is not necessary to use SVD, we hope this presentation is useful to interested readers. Second, UCODE_2014 allows SVD to be used in ways that provide greater insight to users than is commonly supported. For example, insensitive and selected dependent process-model parameters can be held constant during regression instead of the more common practice of holding null-space SVD parameters constant. The latter leads to changes in the values of all process-model parameters which makes it hard for the user to identify process-model parameters for which the observations and prior provide little information. Finally, sensitivity analysis methods that work well for models calibrated with SVD are discussed.

To place the UCODE implementation of SVD and some aspects of this documentation in the context of other works, the following brief literature review is presented along with comments about other selected implementations of SVD.

Using SVD to solve inverse problems has been considered, for example, by Jacobson (1985), Yenihayat (1996), Haber et al. (2000), Tonkin and Doherty (2005, 2009), Doherty (2010), Finsterle and Kowalsky (2011) and Aster et al. (2013). The estimated SVD parameters, which are defined below, are called super parameters by Tonkin and Doherty (2005). The method used here is similar to that used by others, except that a new option for calculating process-model parameter values from SVD parameters is presented and the suggested sensitivity analysis is more extensive than is common.

The SVD capability in UCODE will be suitable for problems in which constraints on process-model parameter values are not applied and when there are few to a moderate number of parameters (generally up to a few hundred parameters). When highly parameterized models developed with methods such as pilot points and Bayesian geostatistics are of interest, the programs PEST (Doherty, 2010; Doherty and Hunt, 2010b), PEST++ (Welter et al., 2012) and bgaPEST (Fienen et al., 2013) provide tremendous capabilities. A program that converts UCODE_2005 input files to PEST is available at the PEST website.

The program iTOUGH2 (Finsterle and Kowalsky, 2011) has a method by which it automatically determines if a model is nonlinear enough that the singular vectors need to be renewed by recalculating process-model parameter sensitivities, and proceeds to recalculate them when needed. Such recalculation is computationally expensive so methods of detecting when it is needed are valuable. With UCODE_2014, the user needs to detect the nonlinearity using methods such as the modified Beale's measure and then begin a new run of SVD to recalcate the sensitivities.

In this work we use the terms "SVD parameters" and "SVD parameter transformation" in describing the SVD theory and methods. These terms are not used, for example, by Aster et al. (2012) and other more mathematically oriented publications. In part the terminology used by others reflects a more fundamental difference in terminology. In the other works listed (and all geophysical literature of which we are aware), the lexicon for using different parameter values is referred to as different models. In the lexicon used in this work, different models are distinguished by using different structural attributes such as boundary condition type, included processes, and parameter structure used to represent spatially and temporally distributed systems properties (such as pilot points, zones, and so on), and best-fit parameters are estimated for each of the different models. To help clearly refer to models that differ in the parameter values (as relevant to SVD) and those that differ in other ways (in the imposed boundary conditions, included processes, parameter structure, and so on), we have found the nonstandard terms "SVD parameters" and "SVD parameter transformation" to be useful. Using this lexicon, we can say that two models differ in the number of SVD parameters estimated for a given defined parameter structure and then also talk about models that differ in other ways. Furthermore, we have found these non-standard terms to be useful when describing SVD to students who are not familiar with the underlying mathematics.

The remainder of this section briefly defines singular value decomposition (SVD) and explains why SVD parameter transformation is used to estimate process-model parameters and how it is accomplished. Then step-by-step instructions for using SVD parameters in UCODE_2014 and a flowchart of the procedure are presented. Finally, a sensitivity analysis for use with SVD is described and discussed along with a computational example.

Definition of Singular Value Decomposition (SVD) Parameters

SVD parameters are defined as linear combinations of process-model parameters; that is, each SVD parameter is calculated by summing terms equal to a coefficient times a process-model parameter value. Thus, SVD parameter a_k can be expressed as

$$a_k = d_{k,1} \times b_1 + d_{k,2} \times b_2 + ... + d_{k,j} \times b_j + ... + d_{k,NP} \times b_{NP} \qquad k=1,NP \qquad (2)$$

where,

NP = the number of adjustable process-model parameters which, by definition, also equals the total number of SVD parameters.

d_{kj} = the elements of singular vector k.

b_j = value of process-model parameter j.

Using matrix equation formatting (for a description of matrix notation see a basic matrix algebra textbook or the web sites listed in Appendix H), equation 2 can be expressed as

$$\mathbf{a} = \mathbf{D}^T\mathbf{b} \tag{3}$$

where **a** is a vector of the NP SVD parameters (a_k) on the left-hand side of equation 2, $\mathbf{D}^T$ is an NP×NP matrix composed of the d_{kj} elements on the right-hand side of equation 2, T takes the transpose of the matrix so that D has elements d_{jk}, and **b** is a vector of the NP b_j process-model parameters on the right-hand side of equation 2.

With SVD, the coefficients d_{kj} are defined in a way that is very advantageous. The advantage occurs because the resulting SVD parameters are all independent of one another – there is no parameter dependence so that all correlation coefficients for SVD parameters are zero by definition. This is accomplished by making each row of matrix $\mathbf{D}^T$ equal the elements of one singular vector. Calculation of singular vectors is described in the following section.

The total number of SVD parameters is the same as the total number of process-model parameters, NP. The NP SVD parameters can be subdivided into SVD parameters that are estimated and those that are not. Estimated SVD parameters are those for which the related singular vectors have larger singular values. The number of estimated SVD parameters is NSVD, and generally NSVD < NP. The number of unestimated (called null-space) SVD parameters equals NP minus NSVD, or what is defined as NPnull in this work.

Calculating Singular Vectors

The coefficients $d_{k,1}$, $d_{k,2}$, …dkj,…$d_{k,NP}$ of equation 2 are the elements of the kth column of matrix **D** in equation 3. In SVD parameter transformation, these columns equal the elements of singular vectors which are calculated by applying SVD to the weighted sensitivity matrix. The description below is derived from Hill and Østerby (2003, Appendices A and B). A detailed treatment of SVD is provided by Golub and Van Loan (1996). A more accessible presentation is provided by Aster et al. (2012).

Any rectangular matrix can be decomposed using SVD. Here, we consider the ND by NP matrix defined as the product of three matrices

$$\boldsymbol{\omega}^{1/2}\,\mathbf{XB} \tag{4}$$

where,

ω is an ND×ND weight matrix that may be diagonal or full. Weight matrices are symmetric and the square root of **ω** needs to be calculated such that it, too, is symmetric.

X is an ND×NP sensitivity matrix with elements equal to $\partial y_i/\partial b_j$, i=1,ND and j=1,NP, where y_i is the simulated equivalent of an observation, prior information, or, possibly, regularization, and b_j is a process-model parameter. The structure of a sensitivity matrix with observations and prior information is shown by Hill and Tiedeman (2007, p. 384). If regularization were included in the regression, it would be added to the **X** matrix in a manner similar to how prior information is added. **X** is evaluated at a given set of parameter values – usually the starting or estimated parameter values.

ND equals the number of rows in sensitivity matrix **X** and it equals the number of observations plus the number of prior information and regularization equations.

B is an NP×NP matrix with the parameter values at which **X** is calculated along the diagonal and zeros off the diagonal. This matrix form is needed for the scaling in equation 4. More commonly in this report, parameter values are listed in a vector referred to as **b**. If a parameter value is less than the value of ScalePval in the Parameter_Data input block of the UCODE_2014 main input file, UCODE_2014 uses the value of ScalePval in **B**. This avoids use of very small values, including values equal to zero.

In many formulations of SVD the scaling by the parameter values is only done when process-model parameters are log-transformed. For log-transformed parameters, the process-model parameter is defined as p_j, b_j is defined as $\ln(p_j)$, $\partial\ln(p_j)/\partial p_j = 1/p_j$, and the sensitivities in **X** are defined using the chain rule as $(\partial y_i/\partial p_j)p_j = \partial y_i/\partial b_j$. The definition of SVD parameters and any SVD-related inference about important and unimportant parameters is different if the parameter is log-transformed or not. Further, if the parameters are not log-transformed the terms of eq. 4 are dimensional.

Including **B** in eq. 4, as is done in UCODE_2014, creates a non-dimensional system of equations for calculating SVD parameters. Thus, results are consistent whether or not parameter values vary over many orders of magnitude, which commonly happens in environmental systems. One concern is that this scaling may not work well for parameters that change with an arbitrary chosen datum, as discussed in the beginning of Chapter 3. For most problems, the issue of parameters that vary over wide ranges is more common than parameters affected by an arbitrary datum, thus **B** is included in equation 4.

The effect of not scaling by the parameter is displayed in the analysis of sensitivity indices reported by Tang et al (2007). In that work, the parameters were not log-transformed and parameter values varied over many orders of magnitude. The local sensitivity results, labeled as PEST results in the paper, are not comparable to global sensitivity methods because the parameter ranges defined in global methods accomplish essentially what the scaling of B accomplishes. That is, it makes the actual parameter value irrelevant and only the range of predictions produced by whatever parameter value range is chosen matters. In Tang et al. (2007), it can be seen that the parameter with the smallest values (LZPK) generally shows up as most important in the PEST results (Tang et al. Tables 3 and 4). The LZPK parameter is not very important in the global sensitivity analysis results

(Tables 5, 6, 7, and 8). LZPK would be less important if the sensitivities were multiplied by the parameter value. This is directly applicable to how sensitivities are scaled in equation 4. Thus, equation 4 is more likely to produce singular vectors that emphasize parameters in a way that would be consistent with global methods.

SVD (singular value decomposition) is used to "decompose" the matrix shown in equation 4, which means that the matrix $\boldsymbol{\omega}^{1/2}$ **XB** is expressed as a product of matrices with specific properties, as follows:

$$\boldsymbol{\omega}^{1/2}\,\mathbf{XB} = \mathbf{USD}^{T} \qquad (5)$$

where,

U is an ND by ND matrix that is orthogonal so that $\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}_{ND}$. $\mathbf{I}_{ND}$ is an identify matrix of dimensions ND by ND; elements on the diagonal equal one and elements off the diagonal equal zero. Each column of **U** is called a left singular vector of $\boldsymbol{\omega}^{1/2}$ **XB**.

D is an NP by NP matrix that is orthogonal so that $\mathbf{D}^T\mathbf{D} = \mathbf{D}\mathbf{D}^T = \mathbf{I}_{NP}$. Each column of **D** is called a right singular vector of $\boldsymbol{\omega}^{1/2}$ **XB**.

S is an ND by NP matrix with elements s_{kk}, k=1,NP equal to non-negative numbers (the singular values) and all other elements of the matrix equal to zero.

For the kth SVD parameter defined by equation 2, the coefficients $d_{k,1}$, $d_{k,2}$, …$d_{k,NP}$ are, by definition, the elements of the right singular vector $\mathbf{v}_k$. Because the right singular vectors are orthogonal, SVD parameters are not correlated. This often makes SVD parameters easier to estimate than process-model parameters, which is one of the reasons that SVD parameter transformation is advantageous.

The other reason that SVD parameter transformation is advantageous is the singular values in the **S** matrix of equation 5. Singular values tend to be large for SVD parameters that can be estimated and small for those that cannot be estimated, and thus provide a convenient way to identify which parameters to estimate. Here we briefly describe some basic properties of SVD; for additional information see Hill and Østerby (2003, Appendix A) and Golub and Loan (1996).

For the matrices of concern in inverse problems no singular value is negative – they are all positive or zero. The decomposition of equation 5 is performed such that the singular values are ordered in **S** from largest to smallest, so that:

$$s_1 \geq s_2 \geq \ \ldots s_{NP-1} \geq s_{NP} \qquad (6)$$

Singular values can be so small that the associated SVD parameter cannot be estimated. In the extreme, small equals zero, but numerical error makes zeros rare in practice. In the absence of any numerical error, if all of the singular values are non-zero, then all NP columns of $\boldsymbol{\omega}^{1/2}$**XB** are linearly independent. This means the NP process-model parameters do not suffer from extreme parameter correlation, and given sufficient sensitivity to the observations (this makes the singular values relatively large) all NP process-model parameters can be estimated uniquely based on the ND observations, prior information, and regularization. In practical application, "small" is larger than zero. Commonly, the number of SVD parameters that have sufficient sensitivity to be estimated equals the number of singular values that are greater than about 10^{-5} to 10^{-6} times the largest singular value. For problems in which process-parameter correlation is not a problem, this is generally consistent with the number of parameters for which CSS values are greater than 10^{-2} times the largest CSS value (Hill and Østerby, 2003).

When some singular values are very small, the matrix is said to be rank-deficient. If the rank of the matrix is considered to be r, r singular values are greater than what is considered to be a small singular value, and the singular values can be expressed as follows.

$$s_1 \geq s_2 \geq \ldots \geq s_r \geq \varepsilon \quad s_{r+1} < \varepsilon \quad s_{r+2} < \varepsilon \quad \ldots \quad s_{NP} < \varepsilon. \tag{7}$$

If the matrix is rank deficient, (NP-r) columns of $\boldsymbol{\omega}^{\frac{1}{2}}\mathbf{XB}$ are linearly dependent. This means that only r parameters can be estimated and a null space of dimension NP minus r is defined. Thus in this work r = NSVD is the number of estimated SVD parameters. Often singular values diminish gradually and there is no clear break between singular values of estimated and null-space parameters. Following the suggestion above, commonly ε is set equal to about 10^{-5} to 10^{-6} times the largest singular value.

In parameter space, this means the following. When r < NP, the right-singular vectors $\mathbf{d}_{r+1}, \ldots, \mathbf{d}_{NP}$ span the null space of $\boldsymbol{\omega}^{\frac{1}{2}}\mathbf{XB}$. Using equation 2, the singular vectors define the linear combinations of the process-model parameters that form the parameter null-space.

When r < NP, then the least-squares solution of **Xb** = **y** is not unique because any vector, **d**, from the null space can be added to the parameter solution **b** without changing the weighted least-squares objective function. Uniqueness requires either adding data (rows of $\boldsymbol{\omega}^{\frac{1}{2}}\mathbf{XB}$; this would mean adding observations, prior information, or regularization), or setting (that is, not estimating) the values of NP minus r process-model or SVD parameters. Setting the values of process-model parameters corresponds to leaving out a column of $\boldsymbol{\omega}^{\frac{1}{2}}\mathbf{XB}$. This could be implemented directly by setting Adjustable=no in the UCODE Parameter_Data input block, but with SVD a different approach is taken. Setting the values of SVD parameters commonly corresponds to not estimating some of the values of s_k of equation 2, specifically those for which the right singular vectors are associated with singular values that are small. In UCODE, this is the default when SVD is used. An unique and useful alternative provided in UCODE_2014 is to assign values to NP minus r process-model parameters and solve for r SVD parameters. These methods of converting from SVD to process-model parameter values are described below.

In practice it is common that singular values span many orders of magnitude fairly continuously. It can be difficult to define the number of parameters to estimate and, therefore, the parameter null space. It generally becomes difficult to estimate SVD parameters with singular values less than about 10^{-5} or 10^{-6} times the largest singular value, so the null space is defined as the singular vectors with smaller singular values. This is consistent with suggestions made by Aster et al (2012), Doherty (2010), and others.

Of interest is how the SVD parameters are composed of the process-model parameters. This can be displayed using stacked graphs of the SVD identifiability (ID) statistic (Doherty and Hunt, 2009). This statistic is described below in the section “ID Statistic and Singular Values”.

Converting from SVD Parameters to Process-Model Parameters

Regression is used to estimate NSVD of the NP SVD parameters. During the estimation process and once estimation is completed, the NP SVD parameters, NSVD of which have been altered in the estimation process, are used to determine values of the NP process-model parameters. The process model is then run to obtain simulated values from which sensitivities and objective functions are calculated.

NP equations are required to solve for the NP process-model parameters from the NP SVD parameters. NSVD of those equations need to be of the form of equation 2. NP minus NSVD is set to the variable NPnull. For the other NPnull equations, two alternatives are included in UCODE_2014. SVDupdate of the UCODE_Control_Data input block allows the user to choose between the two options. The two alternatives are as follows.

(1) Maintain the NPnull **SVD parameters** with the smallest singular values at their starting values (equation 2 evaluated for the starting process-model parameter values). These parameters define what is called the parameter null-space in SVD. This is the most common alternative and is used by Aster (2012, p. 59), PEST and PEST++.
    - A disadvantage of this method is that in general all process-model parameter values change, which can be misunderstood by the user to suggest the existence of significant information in the data on all of the process-model parameters. Yet usually sensitivity analysis and evaluation of parameter correlation indicate that this is not the case. This motivates the second alternative used in UCODE_2014.

(2) Maintain NPnull **process-model parameters** at their starting values.
    - This allows the process-model parameters least informed by the observations to be maintained at their original values. This makes it clear to users that the observations contribute little to estimation of these parameters. The choice of what process-model parameters to set is controlled either by a user-defined priority list or using composite scaled sensitivities (CSS) and parameter correlation coefficients (PCC).

Results of numerical experiments conducted as part of documenting this version of UCODE are reported in Appendix D. For the problem considered, estimated process-model parameters are not affected much by what alternative is used. This makes sense because the process-model parameters set for option 2 are most likely significant components of the SVD parameters set in option 1. The importance of the choice is that by maintaining the values of selected process-model parameters, option 2 results clearly indicate to users which process-model parameters receive no orveery little information from the observations. Having the two options in UCODE_2014 provides a way to investigate their performance in a range of problems.

The following provides a mathematical expression of the two alternatives.

The process-model parameters are calculated from the SVD parameters by solving equation 3 for **b**, which produced the following equation:

$$\mathbf{b}= [\mathbf{D}^{T}]^{-1}\,\mathbf{a} \tag{8}$$

In standard SVD theory, the matrix $[\mathbf{D}^T]^{-1}$ would equal **D** because **D** would have orthogonal columns. However, it is kept in this more complicated form here because option 2 produces a matrix **D** that does not have orthogonal columns.

Consider the typical situation for which the number of SVD parameters estimated, NSVD, is less than the number of defined process-model parameters, NP. Each time the SVD parameters are updated, a_k of equation 2 is determined for NSVD equations and there are NSVD equations that can be used to solve for the NP process-model parameters. Yet NP equations are needed. NP minus NSVD equals NPnull. To define the other NPnull equations needed to solve for all of the NP process-model parameters, the two options above can be considered. $\mathbf{D}^T$ matrices for the two options are illustrated in Figure 5 for a small problem. The two options can be described as follows.

1. **Figure 5a.** Set NPnull unestimated (null-space) SVD parameters, a_k (k=NSVD+1, NP), to their starting values and update all NP process-model parameters. The set NPnull SVD parameters are those associated with the smallest singular values. The starting a_k values are calculated using the starting values of the process-model parameter values (b_j of eq. 2) and the coefficients from the singular vectors ($d_{k,j}$ of eq. 2). Aster et al (2012, p. 59) discuss this approach with respect to linear problems; the approach used here is consistent with their approach as applied to nonlinear problems. This is also the method used in PEST (Doherty, 2010) and PEST++ (Welter et al., 2012) to calculate process-model parameters from SVD parameters.

2. **Figure 5b.** Only update NSVD process-model parameters. Leave NPnull process-model parameters equal to their starting values. The user can decide what process-model parameters to define in the Parameter_Data block using keyword SVDset or UCODE can identify them using CSS and PCC (the method is described in this chapter with the instrutions for the UCODE_Control_Data input block).

(a)

$$\begin{pmatrix} d_{1,1} & d_{1,2} & d_{1,3} & d_{1,4} & d_{1,5} \\ d_{2,1} & d_{2,2} & d_{2,3} & d_{2,4} & d_{2,5} \\ d_{3,1} & d_{3,2} & d_{3,3} & d_{3,4} & d_{3,5} \\ d_{4,1} & d_{4,2} & d_{4,3} & d_{4,4} & d_{4,5} \\ d_{5,1} & d_{5,2} & d_{5,3} & d_{5,4} & d_{5,5} \end{pmatrix}$$

(b)

$$\begin{pmatrix} d_{1,1} & d_{1,2} & d_{1,3} & d_{1,4} & d_{1,5} \\ d_{2,1} & d_{2,2} & d_{2,3} & d_{2,4} & d_{2,5} \\ d_{3,1} & d_{3,2} & d_{3,3} & d_{3,4} & d_{3,5} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure 5. Illustrative examples of matrix $\mathbf{D}^T$ for the two ways of calculating the NP process-model parameter values listed in vector **b** from vector **a** using equation 8. Here, NP=5, NSVD=3, and NPnull=2. For panel (a), vector **a** of equation 8 is composed of SVD parameter values and **D** is completely composed of the elements of singular vectors. The first NSVD=3 SVD parameter values are estimated, the last two remain unchanged – that is, they equal the same SVD parameter values calculated using the starting process-model parameter values. In general, all of the process-model parameter values change during regression For panel (b), vector **a** is composed of the NSVD=3

estimated SVD parameter values and two process-model parameter values. D is composed of the same first three rows, and the last two rows are defined such that the values of the third and fifth process-model parameters remain unchanged.

New Keywords for using SVD in UCODE_2014

SVD parameter transformation is controlled using new keywords in the UCODE_Control_Data, Reg_GN_Controls, and Parameter_Data input blocks.

UCODE_Control_Data Input Block

New keywords have been added to the UCODE_Control_Data input block: SVD, SVDphase, SVDstartpars, SVDnewvectors, SVDnumber, SVDratio, SVDupdate, and SVDprint. Their use is described in this section. If default values for the other new keywords are acceptable (which is often the case), specifying svd=yes is the only addition needed to the UCODE_Control_Data input block.

SVD **Options:** no, yes. **Default=no**

no: SVD parameter transformation is not used.

yes: SVD parameter transformation is used.

The other SVD keywords are used only if SVD=yes. Otherwise, they are ignored. As noted above, often the defaults for these keywords are satisfactory so that none of them need appear in the UCODE_Control_Data input block. The most commonly used additional keywords are SVDratio and SVDnumber.

Once the NP process-model parameters with keyword adjustable=yes in the Parameter_Data input block have been transformed to NP SVD parameters, singular values can be used to determine which SVD parameters can be estimated. Large singular values are associated with SVD parameters that can be estimated. The number of SVD paramters to estimate equals NSVD. The SVD parameters that are not estimated form the SVD null space; there are NPnull=NP - NSVD such parameters. NSVD and NPnull are controlled by the user with keyword SVDratio or SVDnumber.

SVDratio NSVD parameters associated with singular values that are larger than or equal to [SVDratio × (largest singular value)] are estimated. **Default=1x10^{-6}**. The default is used if SVDratio is not entered and SVDNumber is absent or set to an invalid number

SVDnumber NSVD is set to SVDnumber. The NSVD SVD parameters with the largest singular values are estimated. SVDNumber is ignored if a positive value of SVDratio is entered. **Default=0**. Valid numbers are integers equal to 1 through NP, the numbers estimated .process-model parameters

If an SVD UCODE run is repeated using, for example, different SVD control input or another change that would not alter the calculated sensitivities, SVDphase and possibly SVDstartpars can be used to save execution time.

SVDphase **Options:** new, continue. **Default=new**

new: begin with a new UCODE run. This will calculate sensitivities from parameter information in Parameter_Data, use them to calculate singular vectors, and start an SVD assisted parameter estimation given the SVD settings provided by the user in the main UCODE input file, or their defaults if they are not specified.

Continue: continue from a previous UCODE run. This will use parameter values and, usually, sensitivities and singular vectors calculated in a previous UCODE run and saved in data exchange files in the folder from which this continue-UCODE run is launched. An SVD assisted parameter estimation is started given the SVD settings provided by the user in the main UCODE input file, or their defaults if they are not specified. The parameter values and singular vectors used are specified using SVDstartpars and SVDnewvectors. **WARNING:** Data in the data_exchange files will not reflect changes to the process model or UCODE input made by the user in the main UCODE input file subsequent to the initiation of the UCODE run that created the data exchange files.

SVDstartpars Source of parameter values used to begin the simulation.
Options: Parameter_Data, _paopt, _paopt_presvd
Default depends on SVDPhase. If SVDphase = new, by default SVDstartpars = **Parameter_Data**. If SVDphase = continue, by default SVDstartpars = **_paopt**

Parameter_Data: Read values from the Parameter_Data input block of the UCODE main input file. If the Parameter_Values input block is present, values are read from it instead. Use of Parameter_Data is the default for SVDphase=new.

_paopt: Read the most recent values produced by the previous UCODE run. These values are in the _paopt data-exchange file and could be (1) values from the Parameter_Data or Parameter_Values block if the previous run was a sensitivity run; or (2) the most recently estimated parameters if the previous run was a parameter estimation run conducted either with or without use of SVD. Use of _paopt is the default for SVDphase=continue.

_paopt_presvd: Read the values used to start the previous series of regression with SVD parameters. These values are in the _paopt_presvd data-exchange file. This option can be used to start the SVD parameter-estimation from the original parameters using different values for SVDnumber, SVDratio, SVDMaxChange.

Data-exchange files have a filename prefix defined on the UCODE command line and filename suffixes that begin with an underscore, as noted.

The following keyword SVDnewVectors is needed only to continue regression with SVD parameters defined using a new set of singular vectors. The existing SVD parameters can be used (SVDnewvectors=no), or the process-model parameter sensitivities can be recalculated and new coefficients of the SVD parameters can be calculated (SVDnewvectors=yes). The new singular vectors could differ from the old ones for two reasons: (1) For nonlinear problems, sensitivities calculated with the new parameter values are likely to be different from the sensitivities that were calculated using the old parameter values and (2) the parameter values used for scaling in eq. 4 will be different. With SVDnewvectors=yes, sensitivities for all process model parameters are calculated every parameter-estimation iteration so that the time-saving aspect of using SVD is not realized. Indeed, execution times would be longer because sensitivities to SVD parameters would be calculated in addition to sensitivities to process-model parameters. In general, SVDnew vectors = yes is used only rarely to diagnose difficulties in SVD results.

SVDnewVectors **Options:** no, yes. **Default=no**

no: Use original singular vectors calculated with the starting parameters.

yes: Use new singular vectors.

The methods discussed above in the section “Converting from SVD Parameters to Process-Model Parameters” are controlled using the SVDupdate keyword. Briefly, NSVD of the NP SVD parameters are estimated. NP - NSVD=NPnull of the SVD parameters are not estimated. NSVD is determined using keywords SVDratio or SVDnumber as described above in this section. SVDupdate is used to determine if the NPnull parameters are SVD parameters with small singular values or process model parameters. If they are process model parameters, SVDupdate determines how they are to be selected.

SVDupdate **Options:** SVDall, Option2_SVDset, Option2_css_pcc. **Default=SVDall**.

SVDall: all adjustable process-model parameters are updated using SVD parameters. The NPnull null-space SVD parameters are set to the SVD values calculated with the starting process-model parameter values. This is the first option listed in the previous section “Converting from SVD Parameters to Process-Model Parameters”, and has been most commonly used in practice.

Option2_SVDset: some process-model parameters are updated using estimated SVD parameters while the remaining process-model parameters maintain the value assigned at the start of the SVD parameter estimation. The process-model parameters that maintain their original values are assigned based on user-defined values of SVDset, a keyword defined in the Parameter-Data input block description presented later in this chapter.

Option2_css_pcc: some process-model parameters are updated using estimated SVD parameters while the remaining process-model parameters maintain the value assigned at the start of the SVD parameter estimation. The process-model parameters that maintain their original values are assigned based on parameter CSS and PCC values.

For SVDupdate=Option2_SVDset, identification of the NPnull process model parameters that will maintain their starting value are chosen by UCODE with the user-defined value of keyword SVDset in the Parameter_Value input block.

For SVDupdate=Option2_css_pcc, identification of the NPnull process model parameters that will maintain their starting value are chosen by UCODE as the least sensitive and selected highly correlated process-model parameters. These parameters are identified using composite scaled sensitivities (CSS) and parameter correlation coefficients (PCC). The procedure is described in the following text and uses two new keywords, SVDupdate_CSSRat and SVDupdate_CSSAbs. See Appendix E for an example. The list below results in insensitive parameters being included in the set of NPnull parameters before correlated parameters. To include correlated parameters first, set keywords SVDupdate_CSSrat and SVDupdate_CSSabs to zero.

1. The first candidate process-model parameters to be potentially assigned to the set of NPnull parameters are insensitive parameters. These are determined as having a ratio (parameter CSS)/(maximum CSS) that is is less than the value of SVDupdate_CSSrat. The parameter with the lowest ratio is chosen first, and subsequent parameters are chosen with progressively larger ratios. If NPnull parameters are not identified in this step, parameters satisfying a second criteria related to the actual value of CSS are considered.

2. The second set of candidate process model parameters to be potentially assigned as null-space parameters are chosen from the parameters that remain after step 1. The parameters are chosen if their absolute values of CSS are less than SVDupdate_CSSabs, starting with the lowest absolute value. If NPnull parameters are not identified, parameters satisfying a third criteria related to parameter dependence are considered.

3. The third set of candidate process-model parameters to be potentially assigned as being in the set of NPnull set null-space parameters are those for which parameter dependence is high, as measured using PCC. Of concern when considering parameter dependence is that PCC measures the dependence of two parameters but the parameters may be part of a larger set of dependent parameters. For example, if three parameters are dependent, PCC between all of them will be high and generally only one of the three parameters need to be set for the other parameters to be uniquely estimated. UCODE partially addresses this concern by considering sets of dependent parameters with up to three memberes. UCODE begins by identifying sets of three correlated parameters with correlations greater than or equal to SVDupdate_PCC. The parameter with the lowest CSS of the three is selected as a parameter to be set. If after identifying all the sets of three correlated parameters more parameters need to be identified, then parameters other than those involved in the three-way correlations are considered. Of these parameters, pairs of parameters with correlation greater than or equal to SVDupdate_PCC are identified and the parameter with the lowest CSS of each pair is selected.

4. Finally, if additional parameters need to be identified they are selected starting with the lowest CSS of the remaining parameters and moving to higher CSS until the required number of parameters have been selected.

SVDupdate_CSSrat Used to identify null-space parameters when SVDupdate=Option2_css_pcc. See text above to see how it it used. Valid numbers are greater than or equal to 0 and less than 1. Reasonable values are close to 0.0.
Default = 0.01

SVDupdate_CSSabs Used to identify null-space parameters when SVDupdate=Option2_css_pcc. See text above to see how it it used. Valid numbers are greater than or equal to 0.0. Reasonable values are generally near 1.0. **Default = 1.0**

SVDupdate_PCC The value of PCC used to identify correlated parameters when using SVDUpdate=Option2_css_pcc. Parameter correlations with values larger than or equal to SVDupdate_PCC are used as discussed in the steps listed above. Valid numbers are between 0 and 1. Reasonable values are less than but close to 1.0 (e.g. 0.9 to 0.995). **Default = 0.97**

SVDprint **Options:** yes, no. Controls printing of the singular values and singular vectors to the fn.#uout file: No, do not print; yes, print. The singular values and singular vectors are always output to the _svd data-exchange file. **Default=no**.

Reg_GN_Controls Input Block

Three new keywords from the Reg_GN_Controls input block are used when SVD parameter transformation is used. In addition, existing keyword MaxIter is used to define the maximum number of parameter-estimation iterations.

MaxIter When SVD=yes, MaxIter equals the number of parameter-estimation iterations just as it does when SVD=no. **Default=5**.

SVDPerturbAmt a value that controls the fractional amount SVD parameters are perturbed for determining sensitivity. This value applies to all SVD parameters. **Default=0.01**.

SVDMaxChange a value that controls the maximum fractional change allowed within one parameter-estimation iteration for an SVD parameter. This value applies to all SVD parameters. This value is used as described by Hill and Tiedeman (2007, p. 385-389). **Default=2.0**.

SVDTolPar a value that determines when the fractional change in an SVD parameter value is small enough to declare convergence of the parameter estimation process. All parameters need to satisfy this criteria. The same value applies to all SVD parameters. **Default=0.01**.

Parameter_Data Input Block

One keyword is added to the Parameter_Data input block. Constraints defined in the Parameter_Data input block are ignored if SVD=yes.

SVDset an integer used when SVDupdate=Option2_SVDset in the UCODE_Control_Data input block. **Default=0**.

Each parameter is assigned one value of SVDset.

SVDset=0 indicates the value of this parameter will not be set.

SVDset greater than 0 indicates the order in which the process-model parameters are set until NPnull parameters have been selected. NPnull is determined using the SVDratio or SVDnumber keywords in the UCODE_Control_Data input block. The values are always positive integers. A value of 1 indicates the first parameter to set; 2, the second parameter to set; and so on. In the following example, Par3 would never be set, Par1 would be set first, and Par2 would be set second. .

ParName	SVDset
Par1	1
Par2	2
Par3	0

If the user sets SVDset to the same integer for more than one parameter, UCODE terminates with an error message. If there is a gap in the input of integers (e.g. 1 is entered, 2 is skipped and 3 is entered) the gap is closed by decreasing higher values.

The number of parameters that are set by UCODE equals the dimension of the SVD null-space. The dimension of the null-space equals the total number of process-model parameters for which adjustable=yes in the Parameter_Data input block (this number also equals the total number of SVD parameters) minus the number of estimated SVD parameters. The number of estimated SVD parameters is determined using keyword SVDratio or SVDnumber. If the number of parameters with SVDset>0 is less than the dimension of the SVD null-space, UCODE terminates with an error message.

Depending on the input options, it may not be possible to determine the dimension of the null-space until after the sensitivity portion of an SVD run with SVDstartpars=Parameter_Data, so it is often advisable to provide all adjustable parameters a unique value SVDset. Often parameters to be set are either less sensitive (smaller CSS values or first-order effects, for example), or are selected from sets of parameters that are more dependent (absolute values of PCC close to 1.0), or have small t-statistics or total-order effects (measures that combine the effects of sensitivity and dependence).

Flow chart showing how UCODE_2014 performs using SVD parameters

The flowchart for UCODE_2014 when SVD parameters are used is shown in Figure 6.

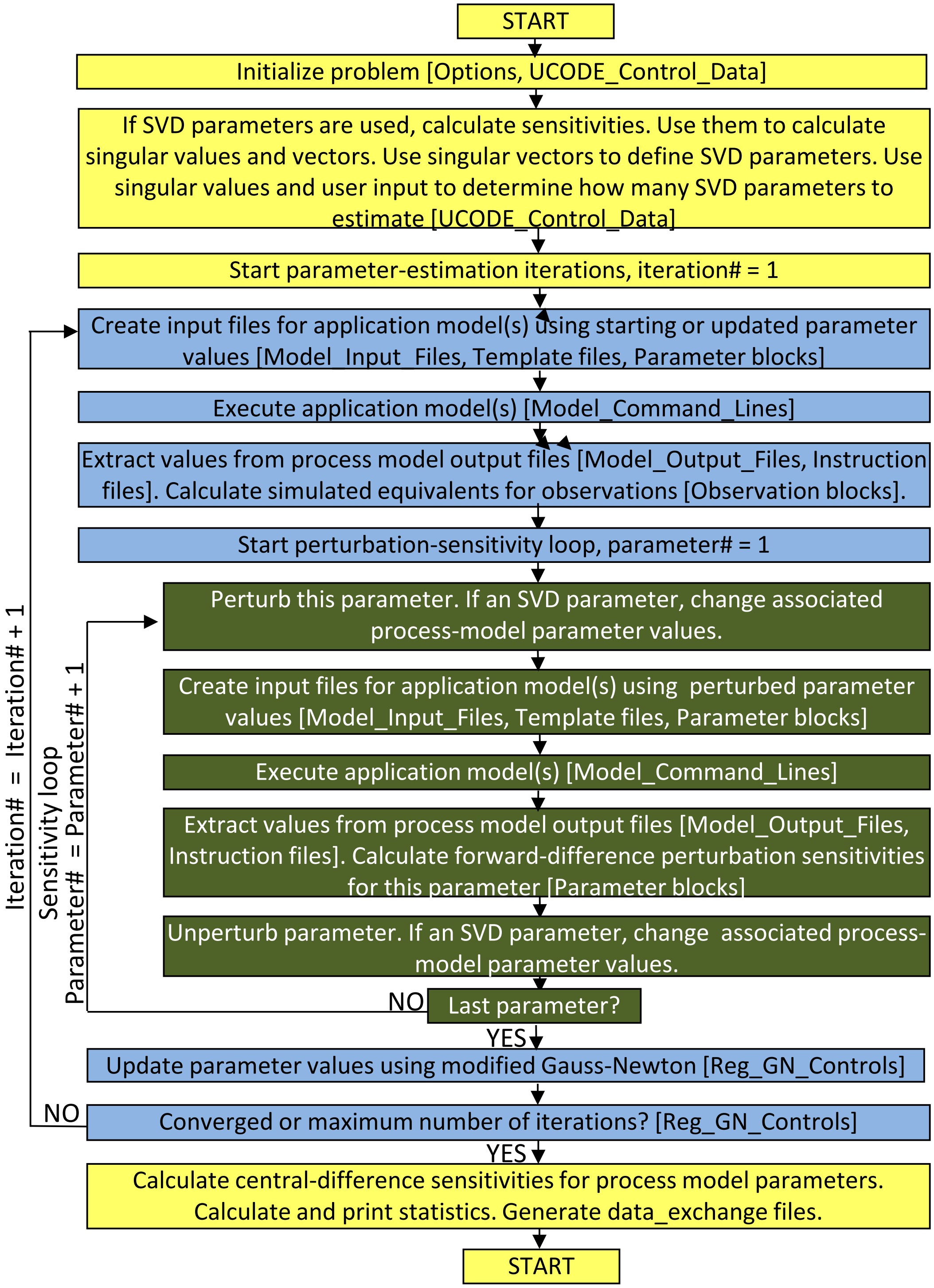

START
Initialize problem [Options, UCODE_Control_Data]
If SVD parameters are used, calculate sensitivities. Use them to calculate singular values and vectors. Use singular vectors to define SVD parameters. Use singular values and user input to determine how many SVD parameters to estimate [UCODE_Control_Data]
Start parameter-estimation iterations, iteration# = 1
Create input files for application model(s) using starting or updated parameter values [Model_Input_Files, Template files, Parameter blocks]
Execute application model(s) [Model_Command_Lines]
Extract values from process model output files [Model_Output_Files, Instruction files]. Calculate simulated equivalents for observations [Observation blocks].
Start perturbation-sensitivity loop, parameter# = 1
Perturb this parameter. If an SVD parameter, change associated process-model parameter values.
Create input files for application model(s) using perturbed parameter values [Model_Input_Files, Template files, Parameter blocks]
Execute application model(s) [Model_Command_Lines]
Extract values from process model output files [Model_Output_Files, Instruction files]. Calculate forward-difference perturbation sensitivities for this parameter [Parameter blocks]
Unperturb parameter. If an SVD parameter, change associated process-model parameter values.
NO
Last parameter?
YES
Update parameter values using modified Gauss-Newton [Reg_GN_Controls]
NO
Converged or maximum number of iterations? [Reg_GN_Controls]
YES
Calculate central-difference sensitivities for process model parameters. Calculate and print statistics. Generate data_exchange files.
START
Iteration# = Iteration# + 1
Sensitivity loop
Parameter# = Parameter# + 1

Figure 6. Flowchart showing major steps in the UCODE_2014 parameter-estimation mode with SVD performance noted. Selected input blocks of the UCODE_2014 main input file and other input files are listed in brackets to help users control the indicated task. ‘Parameter blocks’ represents up to 7 input blocks used to define parameters (Parameter_Groups, Parameter_Data, Derived_Parameters, and the input blocks used to define prior information), ‘Observation blocks’ represents up to 3 input blocks used to define observations. Iteration# is the parameter-estimation

iteration number; parameter# is the parameter number. Colors used to emphasize loops. The central difference sensitivities mentioned at the beginning and end of the flow chart are calculated in a loop similar to that shown for forward difference, but the calculations are repeated for a backward difference as well. The sensitivity loop can produce central difference sensitivities; this is controlled by the user through keyword SenMethod in the Parameter_Data input block.

Understanding Models Calibrated with SVD using Sensitivity Analysis

The input instructions show how to use SVD to estimate parameters in UCODE_2014. SVD provides the opportunity to estimate parameter values that provide an optimized fit to observations with very little understanding by the user about insensitive and dependent parameters and important and unimportant observations. Users can assign SVD=yes in the UCODE_Control_Data input block, use defined default values for all other SVD-related keywords, and probably obtain regression results that fit the observations fairly well with reasonable parameter values. Many of the problems related to non-convergent regression are resolved through SVD because two major difficulties, parameter insensitivity and dependence that results in non-unique parameter estimates, are resolved by a subset of the parameters being quarantined to the SVD parameter null space. For inversions of nonlinear models, the SVD evaluation can be repeated with starting parameter values equal to optimal estimates from a previous SVD run to update the set of parameters that populate the null space.

All users will be pleased to have model calibration proceed so easily. Some will wonder about the results and seek answers to questions like the following. What process-model parameters were most important to this obtained fit? What parameters, and, possibly, related aspects of the model, were not important? What observations are most important? Are any observations that known to be problematic dominating the estimated parameter values? If the model is used for prediction, what are the major contributors to measures of prediction uncertainty and do they make sense? How do I tell if the process-model parameter sensitivities and the coefficients of the SVD parameters need to be recalculated?

Sensitivity analysis can address these questions, and make inverse modeling more transparent and informative. Some useful sensitivity analysis methods are listed in Table 1. Also see, for example, methods and examples presented by Hill and Tiedeman (2007), Saltelli et al. (2208), Doherty and Hunt (2009, 2010a), Foglia et al. (2009, 2013), Hill (2010); Nolan et al. (2011), Borgonovo et al. (2012), Plischke et al. (2013), Rakovec et al. (2014), and Wainwright et al. (2014). Contributors to uncertainty of simulated results is addressed, for example, by Tiedeman et al. (2003, 2004), Saltelli et al. (2008, 2010), and Dausman et al. (2009).

Here we briefly outline an analysis strategy and demonstrate it using results from Hill and Nolan (2011) and Hanson et al. (2012). When using UCODE, output for the analyses are available in data-exchange files after each run of regression whether SVD is used or not.

Only a few of the most common evaluations are presented and demonstrated in this chapter. The statistics are listed in Table 1. Evaluations not covered here include residual analysis, evaluation of model linearity, linear uncertainty associated with predictions, and calculation of correction factors in support of nonlinear uncertainty evaluation. See Table 15 in the UCODE_2005 documentation for the programs used to conduct these evaluations.

Table 1. Five sensitivity analysis methods for understanding models calibrated with SVD.

Name	Definition	Comments and UCODE data-exchange filename suffix	Reference
ID for process-model parameter j	$ID_j = \Sigma_{\ell=1,NPI} (d_{j,\ell})^2$ j=1,NP	_svd_id Sum components from SVD parameters related to parameter j. Summation over all NP SVD parameters (NPI=NP) results in ID=1.0, by definition. Usually presented as bar charts with bars stacked by contributions from each of the NPI SVD parameters.	Doherty and Hunt 2009, 2010a; Hill 2010
Singular Values	Nonzero diagonal entries of **S** in eq. 5. All off-diagonal terms are zero.	_svd Large values identify more dominant SVD parameters	As above, plus Hill and Østerby 2004
t-statistic	$t_j = b_j/(SD_j)$ $= [\mathbf{B}^{-1}(\mathbf{V}(\mathbf{b})/s^2)\mathbf{B}^{-1}]_{jj}\}^{-½}$ $=\{[\mathbf{B}^{-1}(s^2(\mathbf{X}^T\boldsymbol{\omega}\mathbf{X})^{-1})\mathbf{B}^{-1}/s^2]_{jj}\}^{-½}$	b_j and SD_j are listed in _pc.	Regression texts such as Draper and Smith 1998 p. 36-37.
CSS	$CSS_j = \{(1/ND)[\Sigma_{i=1,ND} (DSS_{ij})^2]\}^{½}$ $DSS_{ij} = (\partial y_i'/\partial b_j)\|_{\mathbf{b}}\, b_j\, (\omega_{ii}^{1/2})$	_sc, _sc_svd, _sc_grp Approximately comparable global sensitivity analysis equivalent: first-order effects.	Hill 1998; Hill and Tiedeman 2007; Hill 2010
PCC	$PCC_{jk}=v_{kj}/[v_{jj}^{1/2}\times v_{kk}^{1/2}]$	_pcc, _mc Approximately comparable global sensitivity equivalent: second-order cross-terms.	Draper and Smith 1998. Hill and Østerby 2004. Hill and Tiedeman 2007.

Symbols used in Table 1:

$\boldsymbol{\omega}$, **X**, and **B** are defined for equation 4.

B_j is the value of the jth process-model parameter. Commonly, this value is either the starting value or estimated value of the process-model parameter. It needs to be consistent with the set of process-model parameter values used to calculate the sensitivity matrix **X**.

SD_j is the standard deviation of the jth parameter value. It is calculated using linear theory as the square root of the diagonal of the parameter variance-covariance matrix, $\mathbf{V}(\mathbf{b})= s^2(\mathbf{X}^T\boldsymbol{\omega}\mathbf{X})^{-1}$.

s is standard error of the regression. Its square, s^2, is the calculated error variance of the regression. s^2 is calculated as the value of the least-squares objective function divided by ND minus NP. ND is the number of observation, prior information and regularization; NP is the number of parameters (defined after equations 4 and 2).

V(**b**) is the calculated variance-covariance matrix for the process-model parameters, calculated as $\mathbf{V}(\mathbf{b})=s^2(\mathbf{X}^T\boldsymbol{\omega}\mathbf{X})^{-1}$. Diagonal terms of this NP by NP matrix are variances; off-diagonal terms are covariances. For more information see Hill and Tiedeman (2007, Chapter 7). The approximations required to obtain this equation for **V**(**b**) for nonlinear process models are described by Hill and Tiedeman (2007, Appendix C).

v_{kj} is a term from the parameter variance-covariance matrix, **V**(**b**). If $j\neq k$, v_{kj} is a covariance. If $j=k$, v_{kj} is a variance.

ID statistic and Singular Values

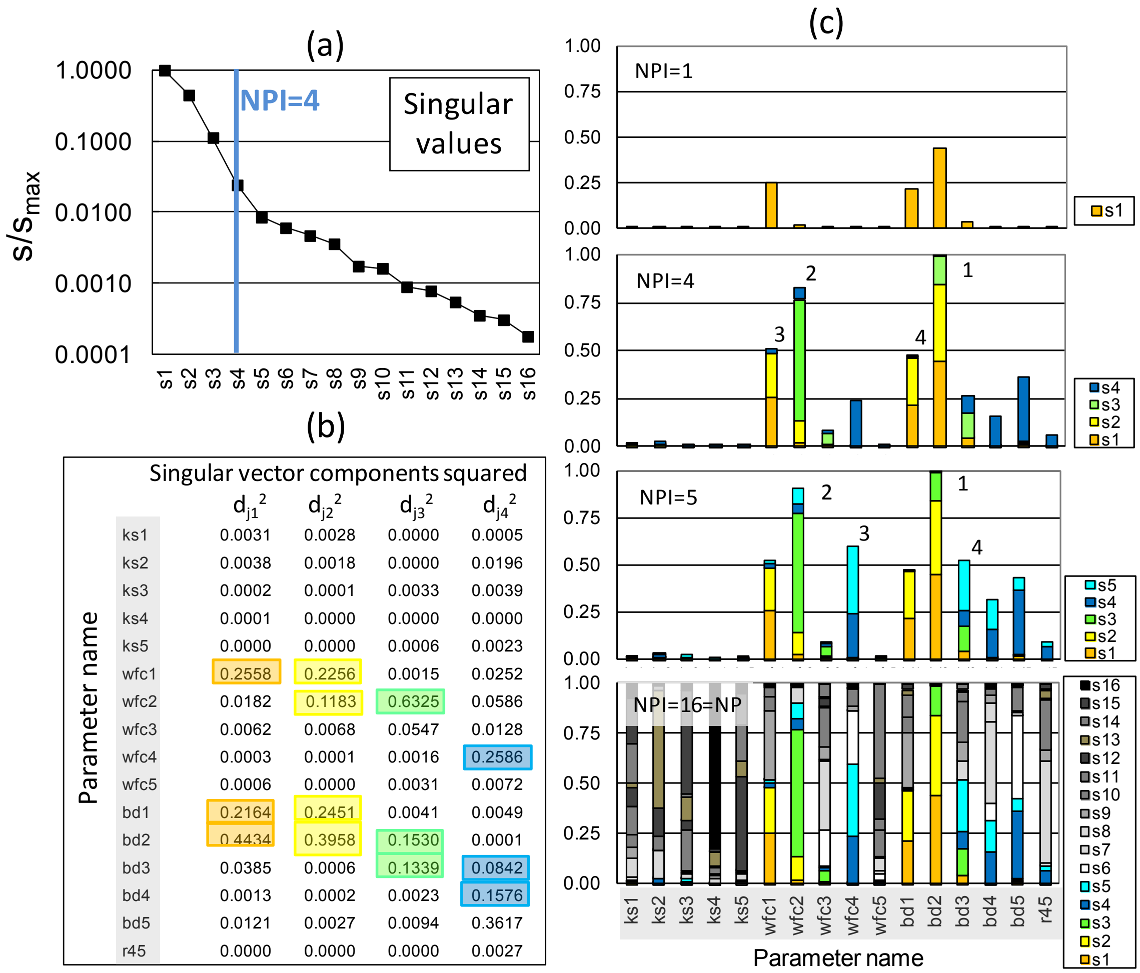
ID statistics (Doherty and Hunt, 2009) clearly display how SVD-transformed parameters are composed of process-model parameters. The importance of the number of SVD parameters included, here termed NPI, is discussed in Appendix E. Figure 7 shows ID graphs for four different values of NPI. The ID statistic is closely related to the composite scaled sensitivity, as shown in Appendix E.

An obvious choice is NPI=NSVD, so that NPI equals the number of estimated SVD parameters. For example, given NPI=NSVD, ID=0.8 for a given process-model parameter indicates that 80% of the process-model parameter is controlled by the estimated SVD parameters; a parameter with ID=0.01 is controlled very little by the estimated SVD parameters.

Commonly ID statistics are presented as stacked graphs such as those shown in Figure 7c. Figure 7c shows ID results for NPI=1, 4, 5, and 16 for a problem described by Hill and Nolan (2011).

Figure 7a also shows a graph of the singular values for this problem. All SVD parameters could be estimated, which is consistent with the range of singular values shown in the graph of Figure 7a being only four orders of magnitude.

Arrows between the table of singular vector components in Figure 7b and the contributions to the ID stacked graphs in Figure 7c highlight how the square of the singular vector components in Figure 7b relate to the stacked graphs of the ID statistic in Figure 7c. The singular vector components in Figure 7b are the same of the coefficients used to define the SVD parameters in equation 2.



Singular vector components squared

Parameter name	d_{j1}^2	d_{j2}^2	d_{j3}^2	d_{j4}^2
ks1	0.0031	0.0028	0.0000	0.0005
ks2	0.0038	0.0018	0.0000	0.0196
ks3	0.0002	0.0001	0.0033	0.0039
ks4	0.0001	0.0000	0.0000	0.0000
ks5	0.0000	0.0000	0.0006	0.0023
wfc1	0.2558	0.2256	0.0015	0.0252
wfc2	0.0182	0.1183	0.6325	0.0586
wfc3	0.0062	0.0068	0.0547	0.0128
wfc4	0.0003	0.0001	0.0016	0.2586
wfc5	0.0006	0.0000	0.0031	0.0072
bd1	0.2164	0.2451	0.0041	0.0049
bd2	0.4434	0.3958	0.1530	0.0001
bd3	0.0385	0.0006	0.1339	0.0842
bd4	0.0013	0.0002	0.0023	0.1576
bd5	0.0121	0.0027	0.0094	0.3617
r45	0.0000	0.0000	0.0000	0.0027

Figure 7. SVD results for the denitrification model of Nolan et al (2010). (a) Singular values divided by the largest singular value, (b) components of the first four singular vectors, which are used to define the first four SVD parameters, and (c) Stacked graphs of the ID statistic, presented for NPI=1, 4, 5, and 16. NP=16 so, ID statistics all equal 1.0 for NPI=16. Arrows show how one squared component of the first singular vector contributes to the stacked graphs of ID statistics. The four graphs on the right are plotted using the _svd_id UCODE data-exchange file. (modified from Hill and Nolan, 2011)

Parameter t-statistic

The equation for the parameter t-statistic listed in Table 2 is scaled to be fit-independent. This is accomplished by dividing out the calculated error variance, s^2, to obtain an expression that does not depend on residuals (observed minus simulated values).

Fit-independent means that the statistic is not calculated using residuals. Fit-independent statistics are useful because they can provide insight even for models that do not fit the observations very well. This is useful, for example, in the beginning of a model development project. Here, fitindependence is achieved by dividing **V**(**b**) by s^2 , where V(b) is defined in Table 1. SD_j/s is referred to as standardized SD in the following discussion. Making the statistic fit-independent does not affect the relative values for a given set of parameters; the same value divides the t-statistics for all parameters of a given model.

Process-model parameters with values of the t-statistic that are large relative to other model parameters are important to the model calibration data. Parameters with large t-statistics have small standardized SD compared to the parameter value. For example, if the t-statistic equals 10, the standardized SD is one-tenth the parameter value; if the t statistic equals 0.1, the standardized SD is ten times the parameter value. Parameters that are important to the observations and prior information are better estimated in that they have smaller linear confidence intervals; that is, values are estimated with considerable certainty. Linear confidence intervals on predictions generally are calculated using parameter values that are plus and minus about two standard deviations from the optimal values.

Parameter values important to observations and prior information, as determined by the t-statistic, may or may not also be important to predictions. The connection to predictions is discussed by Hill and Tiedeman (2007, Chapter 8). Interest in improving the estimation of parameters generally is related to the importance of the parameter to predictions of interest.

CSS and PCC

Parameters identified by the t-statistic as being unimportant to observations and therefore poorly estimated by regression can suffer from insensitivity, parameter correlation, or both. Identifying the likely problem is commonly possible using CSS and PCC.

If SVD parameter transformation is not used, parameters insensitive enough that values probably cannot be estimated can be identified as those with CSS less than 0.01 times the largest CSS calculated for any of the parameters. This applies whether or not the parameter is extremely correlated with any other parameter(s). If SVD is used, parameters with small CSS would be expected to dominate the SVD parameters with small singular values. The user can check this.

If two or more parameters all have PCC absolute values greater than about 0.98, it is likely that one or more of the correlated parameters needs to be set to obtain a tractable regression problem.

If SVD parameter transformation is used, CSS and PCC can be used to understand the singular values and vectors, and the ID statistic because they are closely related, as discussed in Appendix E.

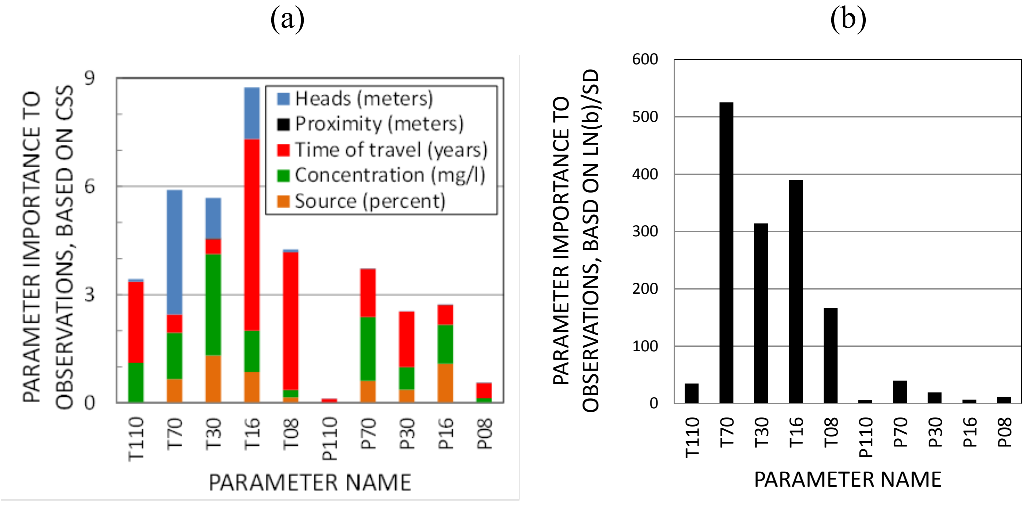


Figure 8. (a) Composite scaled sensitivities (CSS) with bars stacked to reflect contributions from the five kinds of observations available for the test case (see Chapter 6 discussion of constructing plots like this using the _scgrp data-exchange file). (b) t-statistics for the $\log_{10}$ of each of the process-model parameters (obtained using the _pc data-exchange file). For CSS and the t-statistic, large values identify important parameters. CSS is a first-order statistic and ignores parameter interactions (measured by PCC). The t-statistic includes the effects of parameter interactions. It appears that parameter values could be estimated uniquely, as indicated by PCC values all being less than 0.85 and regressions starting from alternative starting values producing similar best-fit parameter values. [Produced using the example model presented in Hanson et al., 2013].

Implications for Monitoring

The sensitivity analysis can provide insight relevant to data collection. Insensitive parameters can only be estimated more precisely by adding more observations, prior information, or regularization directly relevant to the parameter in question.

If correlation plays a significant role, improvement might also be attained by incorporating observations, prior information, or regularization on other parameters.

Large absolute values of PCC indicate that great benefit might be obtained from improving the other parameter values. For example, in groundwater problems it is often possible to improve estimates of a hydraulic conductivity parameter by obtaining new data relevant to one or more recharge parameters because hydraulic conductivity and recharge are commonly correlated. A more quantitative analysis of the worth of additional data can be pursued using the OPR, PPR, and OPA statistics calculated by the computer program OPR-PPR, for which UCODE_2014 provides most of the needed input files.

Uncertainty Evaluation

Whether using process-model parameters or SVD-transformed parameters, uncertainty evaluation needs to account for the effects of both estimated parameters and also uncertainty related to null-space parameters. The latter are generally dominated by insensitive or extremely correlated process-model parameters.

If all prior information on parameters has been included in the regressions conducted for model development, no change will be needed when uncertainty is evaluations. However, often parameters that are ill-informed by the observations, so that they are insensitive and(or) highly correlated, are defined values that are set during model depeloment. These parameters may be important to predictions and need to be activated for prediction simulations. To avoid overestimation of prediction uncertainty, the separate step of adding prior information for these parameters can be very important.Additional discussion about the importance of accounting for uncertainty derived from what are variously called null-space, unestimated, or nonessential parameters is provided, for example, by Anderman et al. (1996), Poeter and Hill (1997), Hill (1998, p. 16-17, 30), Ely et al. (2000), Hill et al. (2001), Hill and Østerby (2003), Tiedeman et al. (2003, 2004), Poeter et al. (2005, p. 26-27, 164, 173), Tonkin et al. (2007, p. 7, 24, 30, 68), Hill and Tiedeman (2007, p. 126, 159, 172, 176, 180-181, 340), Sobol' et al. (2007), Hill (2010), and Doherty (2010). Readers are encouraged to consider those references for additional information.

Chapter 5: MARKOV-CHAIN MONTE CARLO (MCMC) METHODS FOR UNCERTAINTY EVALUATION

Uncertainty analysis is important to using mathematical and numerical models to manage many types of systems, including environmental systems. For example, uncertainty as related to risk assessment is addressed by Borgonovo and Smith (2011), Tartakovsky (2013), and others.

Markov chain Monte Carlo (MCMC) methods are important tools for evaluating uncertainty and performing other types of model analysis when the quantity of concern is not Gaussian distributed (that is, it does not have a normal probability distribution function (pdf)) (Shi et al, 2014). This often occurs for nonlinear models, and is often associated with competing local minima. Here the concept of competing local minima is stressed because unrealistic local minima often do not cause much of a problem. Restarting the regression from another location in parameter space often is all that is needed. In contrast, when many minima provide reasonable results, the problem of local minima is difficult, and global methods like MCMC become essential.

MCMC produces uncertainty intervals for parameters and predictions. The underlying theory is Bayesian so the intervals are called credible intervals. In many circumstances they are similar in size to confidence intervals calculated using the linear or nonlinear methods described by Hill and Tiedeman (2007). See Box and Tiao (1992), Jaynes (1996), Bates and Watts (1980), and Lu et al. (2012) for additional information.

MCMC methods tend to be computationally intensive – often requiring 10,000s to millions of model runs. Linear and nonlinear confidence intervals can be calculated using far fewer model runs. For example, linear confidence intervals can require a number of runs equal to the number of parameters plus one; typically no more than 100 parallelizable model runs. Nonlinear confidence intervals tend to require 1,000s of model runs. For many studies a useful strategy is to use the faster confidence intervals at first and progress to MCMC credible intervals later in the model development process if computer capabilities allow. UCODE_2014 supports such a progressive approach. The relation between MCMC credible intervals and linear and nonlinear confidence intervals is summarized in Table 3. Lu et al. (2012) demonstrate that model adequacy can be more important than the other restrictions.

Table 2 notes that the sum of squared weighted residuals objective function used in UCODE_2014 has an underlying Gaussian error assumption. This comes directly from the functional form of a Gaussian distribution. In MCMC, this means that the likelihood function has an underlying Gaussian assumption. Shi et al. (2014) show that using a Gaussian assumption means that MCMC uncertainty intervals are likely to be closer to the linear and nonlinear confidence intervals produced by UCODE than would occur otherwise. Further exploration of this interesting issue is beyond the scope of the present document.

Table 2. The three types of uncertainty intervals that can be produced using UCODE_2014, computational demands and parallelization opportunities, restrictions, and selected references.

Interval type	Common number of model runs and parallelization opportunity	Restrictions	Reference
Linear	1+ (2×NP). Commonly less than 100. The 2×NP model runs can each be done on separate processors. If there are enough processors, total execution time is equivalent to the time of two sequential forward model runs.	- Process model adequately represents the system.[1] - Model not too nonlinear - Approximately Gaussian, unimodal pdfs for errors in observations, parameters, and predictions.	Hill and Tiedeman (2007)
Nonlinear	Each interval limit requires the equivalent of a full regression run. This is commonly 100s of model runs. For each limit, NP model runs for each nonlinear regression iteration can be done in parallel. Each interval limit is calculated independently of other limits, so each can be calculated using a different set of cores.	- Process model adequately represents the system. [1] - Model not too nonlinear. - Approximately Gaussian, unimodal pdf for errors in observations. - Predictions are an adequately monotonic function of the parameters.	Hill and Tiedeman (2007), Christensen and Cooley (2005), Cooley (2004)
MCMC	10,000s to millions. Chains can each be conducted on different cores. At least 3 and commonly up to NP chains are used.	- Process model adequately represents the system. [1] - Use of a weighted sum of squared residuals objective function in UCODE has an underlying assumption of Gaussian observation errors.	Lu et al. (2012) and references cited therein.

[1] Tests for model adequacy are discussed by Hill and Tiedeman (2007), and many others. Recent studies of model adequacy include Doherty and Welter (2010), Gupta et al (2012), and Foglia et al. (2013).

Example Results

The performance of MCMC was evaluated by Lu et al. (2012). Here we present selected results from that work to demonstrate the utility of MCMC before proceeding to the details of its implementation in UCODE_2014. We use results from two analytical test cases and one groundwater flow problem to illustrate these different types of uncertainty intervals and comment about their utility.

The analytical test cases are not derived from environmental problems; they are easily designed to produce relevant nonlinearities. The first function is designed to be highly nonlinear; the second is mildly nonlinear. The two analytical test cases are defined as follows.

Highly nonlinear test function: $y = x/a + \sin(amx) + \varepsilon$

where a = 2, m = 0.1, and ε is white noise with mean zero and constant variance $s^2 = 1$. Twenty samples of y are first generated with x = {1, 2, …, 20}, and subsequently corrupted using one realization of ε. This nonlinear function has local minima. For this nonlinear function, linear and nonlinear confidence intervals with no prior information and credible intervals with non-informative priors are calculated for y at x = 30 with known s^2.

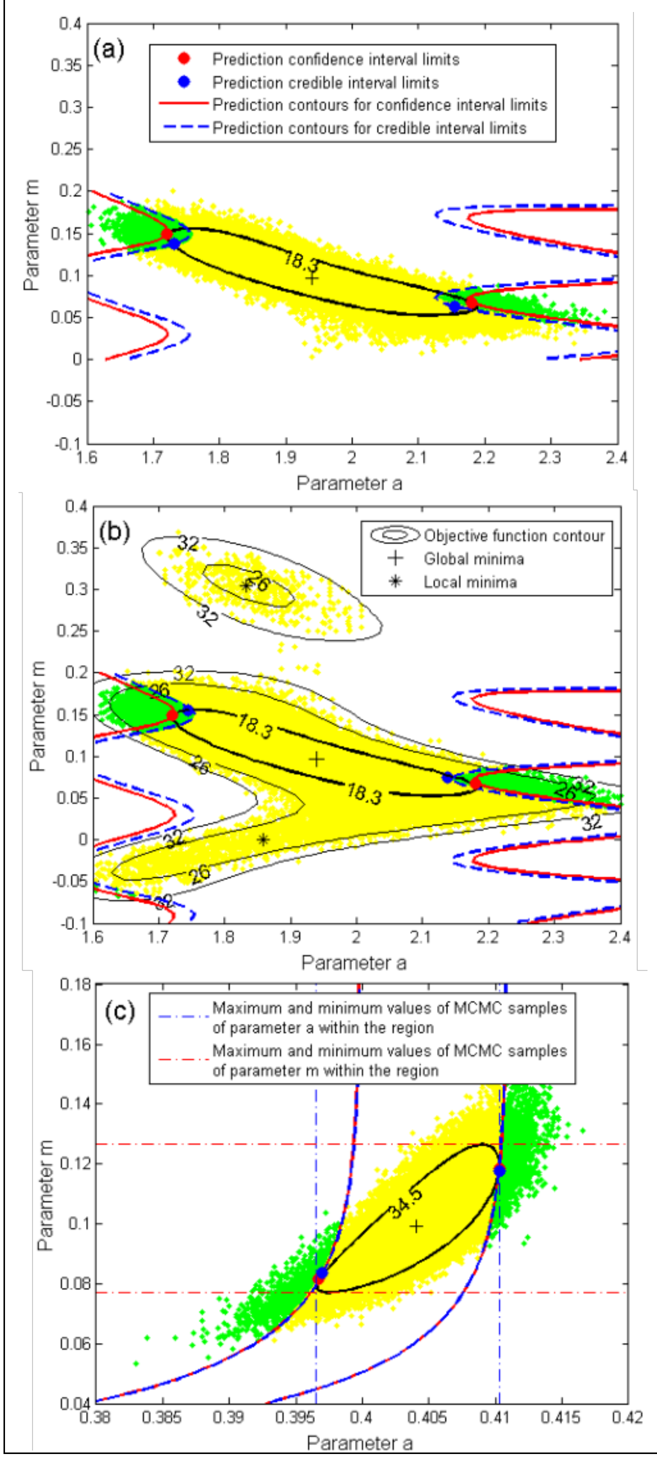
Mildly nonlinear test function: $y = a + (0.49 - a)e^{-m(x-8)} + \varepsilon$

where a = 0.4, m = 0.1, and ε is white noise with mean zero and constant variance $s^2 = 0.01$. Forty samples of y are first generated with x = {8, 9, …, 47}, and subsequently corrupted using one realization of white noise ε. For this nonlinear function, nonlinear confidence intervals and credible intervals are calculated for y at x = 50.

The goal is to identify the values of parameters a and m from the samples of y. The results shown in Figure 9 demonstrate the ability of the MCMC algorithm implemented in UCODE_2014 to explore multiple minima that often occur in highly nonlinear problems.

The groundwater problem is shown in Figure 10, including the true system, which is known for this synthetic test case. Results from three parameterizations of the hydraulic conductivity field are presented: homogeneous (HO), three zones (3Z) and interpolated (INT). See the caption of Figure 10 for comments about these parameterizations. Figure 11 shows uncertainty intervals calculated in the three ways available in UCODE_2014. The different types of intervals require very different computational effort. These results suggest the following:

1. None of the uncertainty methods work well when the model is too much in error, as for the HO model. However, even the erroneous model performed reasonably well with the 18 streamflow observations. Unfortunately, in general groundwater contributions to streams cannot be measured with the accuracy needed to use such streamflow observations given existing technology.
2. Given the differences in predicted values between the three models, the difference in uncertainty intervals seems rather modest. This suggests that investing in considering alternative models can be more important than investing in considering more computationally intensive uncertainty measures.
3. Measures of intrinsic nonlinearity are of some value in identifying when more computationally demanding uncertainty intervals are needed.



EXPLANATION

🟡 Realization from the middle 95% of generated values

🟢 Realization from the top and bottom 2.5% of generated values

🔵 Limit of a credible interval from MCMC sampling calculated as the 2.5% or 97.5% value. For 1,000 realizations, that would be value 25 and 975 from the ordered set of 1,000 values.

− Contour line of the sum of squared weighted residual objective function. Each point along the line is calculated using the parameter values that can be read from the two graph axes. Thus, the range of values that can produce the same model fit can be determined from the figure.

🔴 Limit of an individual nonlinear confidence interval, calculated as the extreme high and low points of an objective function contour. Here the values in (a) and (b) of 18.3 and in (c) of 34.5 were calculated using eq. 8.14 of Hill and Tiedeman (2007) with a student-t critical value; this is the same as eq. 16 of Lu et al. (2012).

– –Solutions of the function with the same simulated value as that produced at the uncertainty interval limit.

Figure 9. The 95% individual nonlinear confidence and credible intervals for (a and b) the very nonlinear test function (dots in (a) are samples from MICA of Gallagher and Doherty, 2007; dots in (b) are from DREAM used in UCODE_2014) and (c) the mildly nonlinear test function (from Lu et al., 2012).

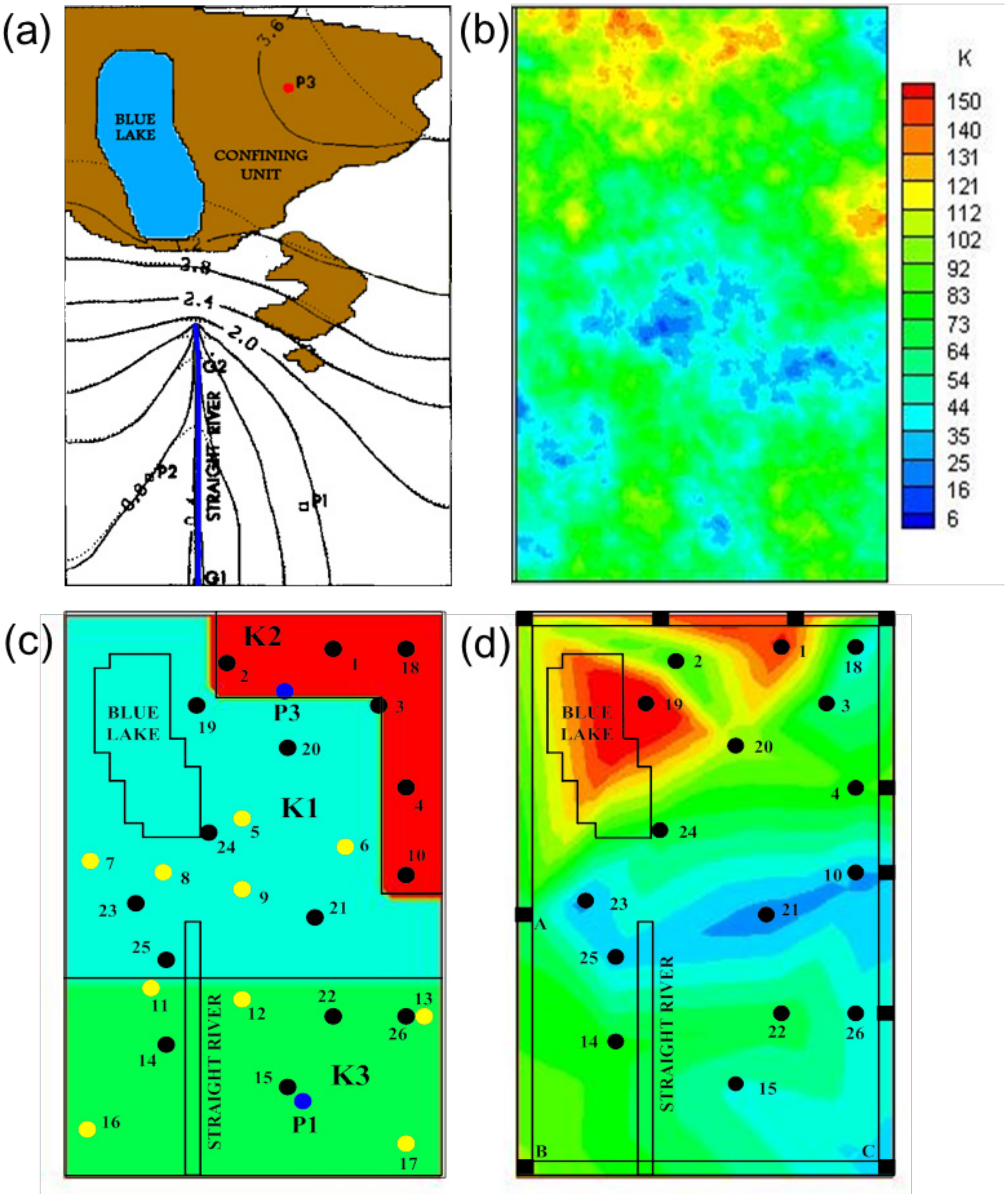


Figure 10. (a) Modeling domain with the confining unit, Blue Lake, and Straight River, (b) the true horizontal hydraulic conductivity field (K field) with values from 6 to 150 m/d, and the calibrated K fields for (c) the three-zone model (3Z) with K2 up to 317 m/d and (d) the interpolation model (INT) with highest value 180 m/d. Panel (d) shows dots upon which linear triangular basis functions are constructed to interpolate the K field for INT. The hydraulic conductivities are estimated by regression at the sixteen numbered inner dots and at the three squares labeled A, B and C in (d); the hydraulic conductivities of the other seven squares without labels are not estimated. In (c) dots represent 27 wells where hydraulic heads are observed at the top and bottom of the system. Black dots (same as those in (d)) also have hydraulic conductivity measurements and yellow dots only have head observations; pumping occurs at wells P1 and P3 (blue dots), and drawdown is predicted at well P3 (from Hill et al., 1998; Lu et al., 2012).

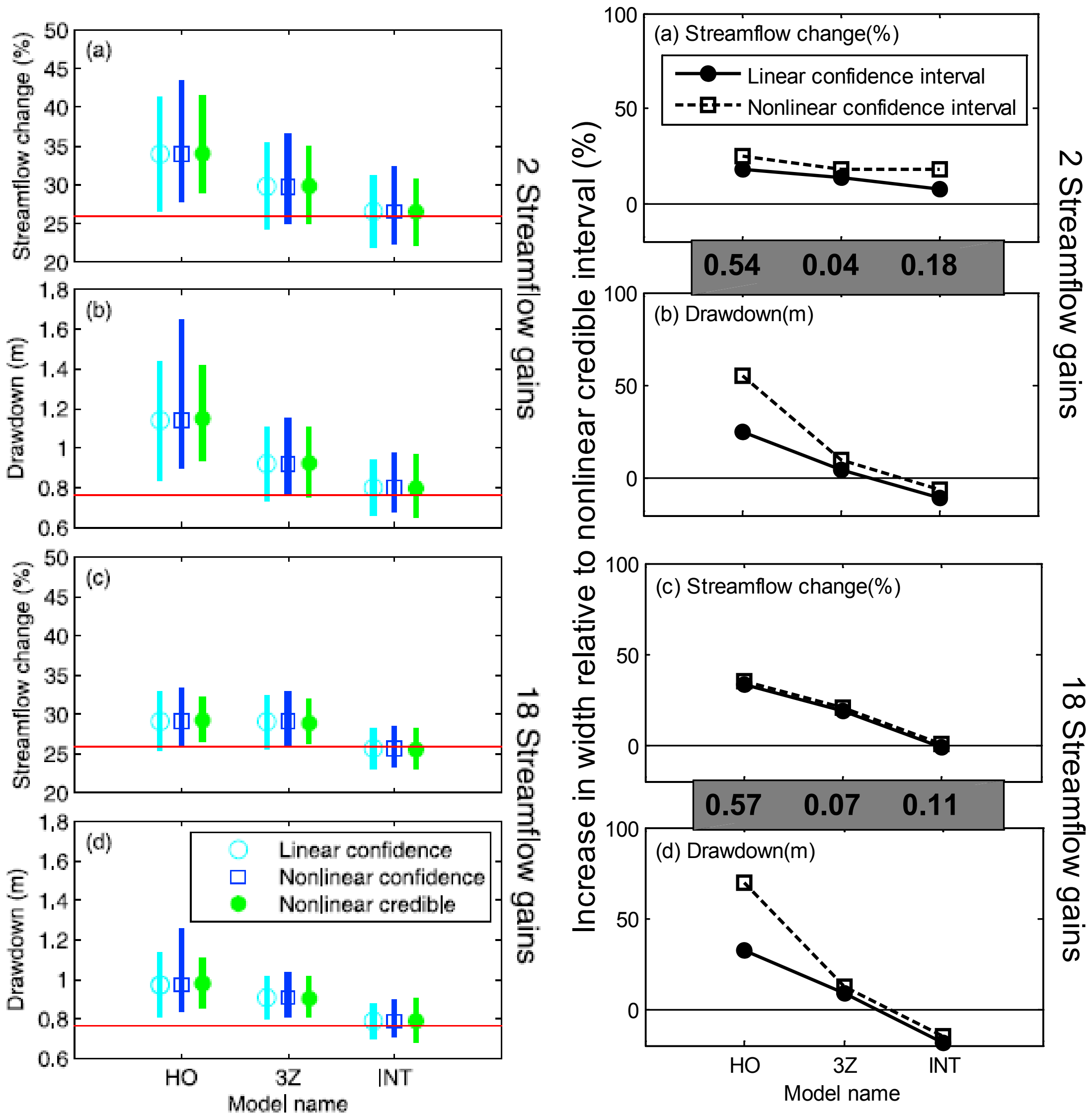


Figure 11. Left column: Linear and nonlinear confidence intervals and nonlinear credible intervals (≤106, ≤1594, and 420,000 model runs, respectively). The red lines are the true values of the predicted quantity, which are known for this synthetic problem. Right column: Relative increase in width of linear and nonlinear confidence intervals to nonlinear credible intervals and measures of intrinsic nonlinearity. Results are presented for two predictions: (a) and (c) streamflow change at gauge site G2 and (b) and (d) drawdown of the water table at well P3. Here 2 or 18 observations of streamflow gain are used. In the left column, the horizontal lines represent true values of the predictions. The nonlinear credible intervals are calculated using the DREAM algorithm implemented in UCODE_2014.

MCMC Procedure and Flowchart

The flowchart in Figure 12 shows the steps for MCMC as implemented in UCODE_2014.

The MCMC runs are composed of chains – sequential sets of parameter values. In Figure 12, the steps starting at the second box and ending at the second to last box are conducted for each chain. Each parameter set depends on previous sets of values, which complicates parallelization of the MCMC algorithm. Parallelization can be accomplished by running multiple chains simultaneously on separate cores.

MCMC typically is used to calculate the posterior probability distribution function (pdf) of the parameters and resulting pdf of predictions. It uses Bayes equation, which is expressed as (Papoulis, 1991)

$$p(\mathbf{b}|\mathbf{y}) \propto L(\mathbf{b}|\mathbf{y})p(\mathbf{b}) \tag{9}$$

where $p(\mathbf{b}|\mathbf{y})$ is the posterior distribution of model parameters **b** given the observations **y**, $\propto$ means proportional to, $p(\mathbf{b})$ is the prior pdf on the parameters, and $L(\mathbf{b}|\mathbf{y})$ is the likelihood of the parameters given a set of observations. $L(\mathbf{b}|\mathbf{y})$ is equivalent to $p(\mathbf{y}|\mathbf{b})$, the probability of the observations given the parameters.

In this work the likelihood function is constructed assuming Gaussian errors in the observations, which leads to the weighted least-squares objective function used by UCODE. The likelihood term in equation 9 includes only contributions from observations. For nonlinear problems, local minima can occur in the objective function surface and Gaussian observation errors do not tend to produce Gaussian errors in the predictions, even if the prior is Gaussian. MCMC is important for quantifying uncertainty in problems characterized by local objective function minima that compete with the global minima and significantly non-Gaussian prediction pdfs, such as multi-modal pdfs.

For the prior pdf on the parameters, UCODE allows the user to select from 11 distributions. Ten of these distributions are for individual parameters: uniform, normal (also called Gaussian), log-normal, beta, chi-square, inv-chi-square, scaled-inv-chi-square, gamma, inv-gamma, and exponential. Dependence between the parameter in the prior pdf is allowed only if a multivariate normal (Gaussian) distribution applies.

The first parameter sample in each Markov chain is often drawn randomly from the prior distribution of parameters specified by the user. Alternatively, if a regression or sensitivity analysis run is available, it can be used to initialize the chain by setting flag UseRegResult=yes in the MCMC_Controls input block described later in this chapter. In this situation, the first parameter sample is drawn from a multinormal distribution with a variance-covariance matrix calculated based on linear assumptions, using a vector of mean values set to optimal parameter estimates read from the _paopt file and a parameter variance-covariance matrix read from the _mv file. The _paopt and _mv are filename suffixes for a file produced by UCODE. The prefix of the filename is defined on the UCODE command line. Our computer experiments show that for a unimodal problem, using regression results to initialize the chain can speed up convergence, resulting in MCMC converging within 1,000s of iterations (with one model run per iterations). Multimodal problems using regression results to initialize the chain may need a large number of runs to fully define the other modes and obtain MCMC convergence.

After initializing the chain, subsequent parameter samples are generated based on the DREAM algorithm of Vrugt et al. (2008, 2009), which was coded in Fortran for UCODE_2014. A candidate sample is generated by the difference between current samples of different chains, and then the candidate sample is accepted or rejected using the Metropolis rule. The MCMC simulation ends when the number of generated parameter samples achieves the amount set by the user through keyword MaxSamples. If the MCMC simulation ends without achieving convergence, more MCMC iterations are needed. Convergence is diagnosed using the Gelman-Rubin criteria (Gelman and Rubin, 1992). The simulation is thought to be not converged if the calculated value is greater than the threshold set by the user through keyword GelmanR. In this case, the user can continue the simulation by setting flag Restart=yes in the MCMC_Control_Data input block (see description below in this chapter). Then, a new MCMC run will continue from the place at which a previous MCMC run terminated.

Thus, the procedure for an MCMC run can be described as follows, where additional detail is provided for starting chains:

- Start the chain (second box in Figure 12) in one of three ways.
    - Start with a UCODE regression or sensitivity analysis run. If regression runs have been conducted with UCODE, uncertainty evaluation may require activating any parameters that were set during the regression, or for which prior information was defined to be more dominant than independent information justifies. See discussion in Hill and Tiedeman (2007, p. 131). This requires conducting a UCODE sensitivity mode run to produce _paopt and _mv files with consideration of all parameters that may be used in the next step.
    - Start with a MCMC run directly with first parameter samples drawn from the prior distributions defined in the MCMC_Prior_PDF input block.
    - If MCMC terminated without satisfying the Gelman-Rubin criteria specified using keyword GelmanR in the MCMC_Control_Data input block, start a new MCMC run using the final values from the previous run.
- Run UCODE in MCMC mode, which does the following as shown in Figure 12.
    - Initialize MCMC.
    - Proceed through Markov chains using the DREAM algorithm. After the burn-in period (convergence is diagnosed using the value defined using the GelmanR keyword), generate parameter samples from the parameter posterior probability density function. The quality of the posterior distribution is determined by the number of MCMC runs conducted.
    - Simulate predictions for all generated parameter samples after convergence by setting flag MCMC_PREDICTION=yes.

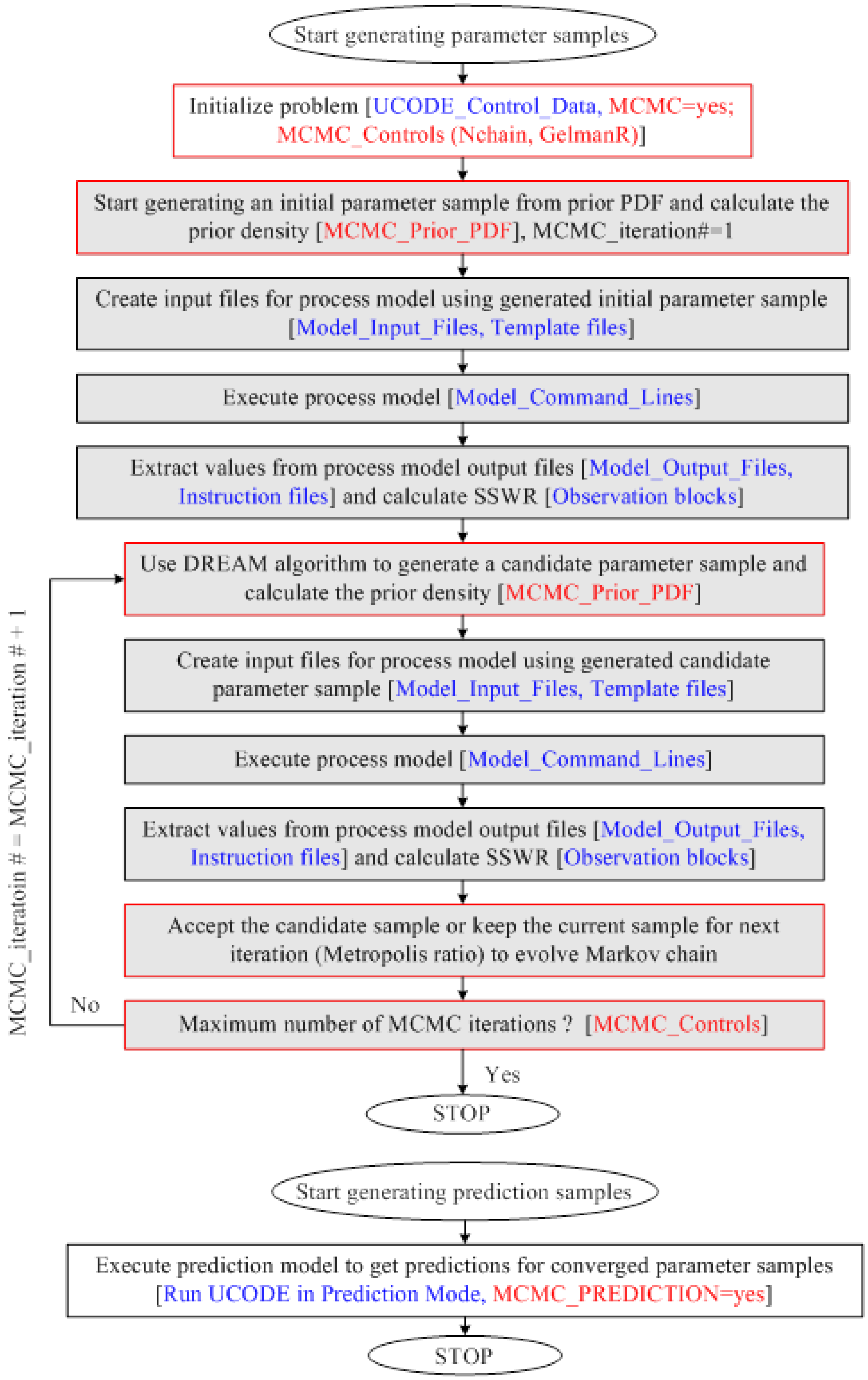

Start generating parameter samples
Initialize problem [UCODE_Control_Data, MCMC=yes; MCMC_Controls (Nchain, GelmanR)]
Start generating an initial parameter sample from prior PDF and calculate the prior density [MCMC_Prior_PDF], MCMC_iteration#=1
Create input files for process model using generated initial parameter sample [Model_Input_Files, Template files]
Execute process model [Model_Command_Lines]
Extract values from process model output files [Model_Output_Files, Instruction files] and calculate SSWR [Observation blocks]
Use DREAM algorithm to generate a candidate parameter sample and calculate the prior density [MCMC_Prior_PDF]
Create input files for process model using generated candidate parameter sample [Model_Input_Files, Template files]
Execute process model [Model_Command_Lines]
Extract values from process model output files [Model_Output_Files, Instruction files] and calculate SSWR [Observation blocks]
Accept the candidate sample or keep the current sample for next iteration (Metropolis ratio) to evolve Markov chain
Maximum number of MCMC iterations ? [MCMC_Controls]
MCMC_iteratoin # = MCMC_iteration # + 1
No
Yes
STOP
Start generating prediction samples
Execute prediction model to get predictions for converged parameter samples [Run UCODE in Prediction Mode, MCMC_PREDICTION=yes]
STOP

Figure 12. Flowchart of MCMC capability in UCODE_2014 with the parameter generation UCODE run (called the MCMC run in the text) at the top and the prediction UCODE run (called the MCMC_Prediction run in the text) on the bottom. MCMC-specific parts are shown in red and UCODE input blocks other than those unique to MCMC shown in blue. Input blocks with keywords that control performance are listed in brackets. The

boxes with gray shading are repeated in each chain, where each chain is generally computed using a different core, thus forming a parallelization of MCMC.

New Keywords and Input Blocks for MCMC

Two new keywords in the UCODE_Control_Data input block and three new input blocks are used to implement MCMC. New input block MCMC_Controls controls the MCMC simulation. New input blocks MCMC_Prior_Groups and MCMC_Prior_PDF define the prior distribution.

UCODE_Control_Data Input Block (required)

Two keywords are **added** to the UCODE_Control_Data input block.

MCMC yes/no. Yes activates the MCMC capability to generate parameter samples which are saved in files with filename suffix._mcmc_par**, where ** is replaced by the chain index, i.e., 01, 02. **Default=no**.

MCMC_PREDICTION yes/no. Yes activates the MCMC capability in prediction mode to simulate prediction samples which are saved in files with filename suffix._mcmc_pred**, where ** is replaced by the chain index, i.e., 01, 02. When MCMC_PREDICTION=yes, keyword MCMC also needs to be yes for MCMC prediction results to be calculated. **Default=no.**

In some cases correct weighting of observations requires that Use_wt=yes in the UCODE_Control_Data input block (see Chapter 3 of this report). Use_wt=yes results in observation weights being read from the _wt data-exchange file instead of being calculated using input in the Observation_Data input block. The values in the _wt file will be different if the final weights are calculated using simulated values. Regression linear confidence intervals are calculated based on the final weights, which are listed in the _wt file at the end of a regression run of UCODE. To be consistent with the regression interval calculation, the weights used in the likelihood function calculation in MCMC should be read from the _wt file.

MCMC_Controls Input Block (optional)

MCMC_Controls input block supports eleven keywords by which performance of MCMC can be managed. All keywords have default values that are used if this input block is not found, as occurs if the input block is missing or the name is misspelled.

The only keyword for which the value is not used when MCMC=yes is ItStartPred. When MCMC_PREDICTION=yes, four of the keywords are used: Nchain, MaxSamples, PrintStep and ItStartPred. Unused keywords are ignored.

Nchain Number of Markov chains (for the DREAM algorithm to work, make Nchain=3 or greater than half of the number of parameters, whichever is largest). **Default=5.**

Chains can be run in parallel using the processors defined for the UCODE parallel processing capabilities documented in Chapter 12 of the UCODE_2005 documentation. If enough processors are available, it is suggested that Nchain equal the number of parameters. When MCMC_PREDICTION = yes, Nchain must be specified if it was specified for the MCMC=yes model run used to generate the parameter values. For predictions, Nchain determines the number of parameter sample chains that are used to make predictions. It will generally equal the value set for the MCMC run, and cannot be larger. If a smaller value is specified in the MCMC_Prediction run, only some of the MCMC generated parameter values are used.

MaxSamples Maximum number of samples generated for each chain, including the burn-in period. To help obtain accurate results, **Default=10000**.

In the example distributed with UCODE_2014 a small value of MaxSamples=20 is specified to allow users to quickly see the logistics of an MCMC run. Increase MaxSamples for all applications of MCMC.

When MCMC_PREDICTION = yes, MaxSamples is used as follows. The parameter samples of each chain between iteration ItStartPred+1 and MaxSamples are used to make predictions. MaxSamples specified for an MCMC_PREDICTION run should not be larger than the value set for the associated MCMC run and needs to be larger than the value of ItStartPred defined below.

The following three keywords, Npair, NCR, and JumpStep, are three parameters of the DREAM algorithm (Vrugt et al., 2008, 2009), and the values chosen affect the Markov chain convergence. Npair represents the number of pairs of chains used to generate candidate samples; NCR is the number of CR (crossover probability) values used to determine the subspace sampling strategy; and JumpStep defines the step interval after which the jump rate is set to a large value to enable jumping between different modes.

DREAM accelerates chain convergence in the following ways:
(1) DREAM generates a candidate sample using a differential evolution offspring strategy. It uses more than one pair of chains, which increases the diversity of the proposal and thus facilitates the updating of the proposal distribution to the target;
(2) DREAM only updates selected dimensions (i.e., subspace sampling strategy) when a candidate sample is generated. This improves efficiency for high-dimensional problems because with increasing dimensions it is often not optimal to change all dimensions simultaneously;
(3) DREAM adjusts the jump rate to a large value in a high frequency. This increases the chance of jumping between different modes and speeds up convergence.

The above three ways of accelerating convergence are related to the three parameters, Npair, NCR, and JumpStep, respectively. Vrugt et al. (2009) shows that a value of Npair greater than 1 is slightly better and Npair=3 works well for most problems. NCR value should be larger than 1 to make the subspace sampling strategy more active. Vrugt et al. (2009) demonstrates that NCR=3 works well for most problems. JumpStep=5 is a good choice to adjust the jump rate; a higher value may reduce the chance of jumping between modes; a lower value may cause a small acceptance rate.

The default values are heuristic. No theories indicate a best value of the three parameters. But many practical tests show that Npair=3, NCR=3, and JumpStep=5 works well for most problems.

Npair Number of pairs of chains to generate candidate samples. The δ value defined in equation (4) of Vrugt et al. (2009). **Default=3**.

NCR Number of CR values which determines the subspace sampling strategy in DREAM algorithm. The n_{CR} value used in Vrugt et al. (2009). **Default=3**.

JumpStep Step interval to reset jump rate to enable jumping between different modes of the posterior pdf. **Default=5**.

UseRegResult yes/no. Yes uses the multinormal distribution from the end of a parameter estimation run to start MCMC. **Default=no**.

GelmanR Gelman-Rubin convergence diagnostic criterion. The value specified is inter-chain, so one value for each parameter is calculated regardless of the number of chains. **Default=1.2**.

If the calculated Gelman-Rubin values are all smaller than the value of GelmanR, it is assumed that at that iteration MCMC has converged. The calculated Gelman-Rubin values are saved in a data-exchange file with filename suffix _mcmc_grr. The iteration numbers at which the calculated Gelman-Rubin value is smaller than GelmanR are printed in the main output file of MCMC run with filename suffix #umcmc. Use the information from either file to determine convergence. After this iteration, all generated parameter samples can be used to make predictions, and the keyword ItStartPred defined below should be set to a value no smaller than this iteration number.

PrintModelOutput yes/no. Yes prints the corresponding simulated values of observations based on the parameter samples. The simulated values are saved in files with filename suffix._mcmc_sim**, where ** is replaced by the chain index, i.e., 01, 02. **Default=no**.

PrintStep Step interval to print calculated Gelman-Rubin values in data-exchange file with filename suffix _mcmc_grr, to print parameter samples in files with filename suffix._mcmc_par** and to print simulated values in files with filename suffix ._mcmc_sim**. **Default=10**.

When MCMC_PREDICTION = yes, set PrintStep to the same value used in the associated MCMC run.

Restart yes/no. Yes restarts a new MCMC run at the place at which a previous MCMC run terminated. The samples are saved in the data-exchange file with filename suffix _mcmc_restart. **Default=no**.

If the previous MCMC run converged as indicated by word 'T' printed at the end of the file with suffix _mcmc_restart, all the generated parameter samples in the new MCMC run can be used and the Gelman-Rubin convergence information is not printed. If the previous MCMC run did not converge as indicated by word 'F' printed at the end of the file _mcmc_restart, the printed Gelman-Rubin information still needs to be used to diagnose the convergence of the new MCMC run.

ItStartPred MCMC iteration number after which the parameter samples are used to simulate predictions. Its value should be set as the iteration number at which chain convergence is reached or a larger value. This variable is only used when MCMC_PREDICTION = yes. **Default=0.5*MaxSamples**, where MaxSamples is the value set in MCMC prediction run.

An example MCMC_Controls input block showing what keywords are used when in the UCODE_Control_Data input block MCMC=yes and MCMC_PREDICTION=no by designation or default. Unused keywords are ignored.

```
BEGIN MCMC_CONTROLS KEYWORDS
#Used for all MCMC runs
Nchain=6
MaxSamples=10000
PrintStep=10
Npair=3
NCR=3
JumpStep=5
UseRegResult=no
GelmanR=1.2
PrintModelOutput=yes
Restart=no
#The following is only used when MCMC=yes and MCMC_PREDICTION=yes
ItStartPred=5000
END MCMC_CONTROLS
```

MCMC_Prior_Groups Input Block (optional)

The keywords defined in the MCMC_Prior_Groups and MCMC_Prior_PDF input blocks may all be needed when MCMC=yes. When MCMC_PREDICTION=yes, only the keyword ParamName defined in the MCMC_Prior_PDF block is used. In both cases, the defaults can be used and this input block is always optional.

Keywords in this input block include:

GroupName Name for a group of parameters (up to 12 alphanumeric characters including underscore, _, not case sensitive). Default=DefaultPrior. All parameters for which a multinormal distribution is defined need to be in the same group.

CovMatrix Name of the variance-covariance matrix of the group of parameters to which all parameters with a multinormal prior distribution need to belong. The matrix is specified in the Matrix_Files input block as described in Chapter 10 of Poeter et al. (2005). CovMatrix only needs to be specified when PDFType is multinormal. For groups with other designations of PDFType, CovMatrix is not used. If table format is used for the MCMC_Prior_Groups input block, list CovMatrix as the last keyword in the table and for all other groups leave the field blank as shown in the example after the description of the MCMC_Prior_PDF input block. Please see more on CovMatrix just after Other keywords are discussed.

Other keywords Any keyword from the MCMC_Prior_PDF input block. Keyword values defined here are replaced if values are defined in MCMC_Prior_PDF.

CovMatrix is needed when the prior distribution of a group of parameters is specified as multinormal. Only one variance-covariance matrix can be defined for all parameters for which prior pdfs are designated as multinormal. All parameters with multinormal distributions need to be assigned this GroupName and a block covariance matrix needs to be defined for the group using keyword CovMatrix.

For listed parameters to be normally distributed and independent, do one of the following.

(1) In the MCMC_Prior_PDF input block, define PDFType as normal and define the standard deviation using PDFVar2.
(2) In the MCMC_Prior_PDF input block, define PDFType as multinormal and define the CovMatrix as diagonal for the independent parameters. This is accomplished by specifying nonzero values on the diagonal of the matrix, which are variances, and zeros for all off-diagonal terms, which are covariances. This setup makes it easy to include correlations of some of the parameters – simply make some of the off-diagonal terms non-zero. If the correlated parameters are listed together in the MCMC_Prior_PDF input block, and parameter dependence occurs only between parameters within subsets, CovMatrix will be a block diagonal matrix.

CovMatrix is defined by the user in a file read using the Matrix_Files capability described in Chapter 10 of the UCODE_2005 documentation. The names of the parameters having multinormal distribution and their mean values are defined in the MCMC_Prior_PDF input block. The first parameter defined there as mutinormal is associated with the first row and column of CovMatrix, the second is associated with the second row and column, and so on.

If Blockformat KEYWORDS is selected by designation or default, keywords associated with a prior PDF group in the MCMC_Prior_Groups input block need to follow the related GroupName either on the same line or a new line. The GroupName keyword needs to be the first keyword on a new line.

If Blockformat TABLE format is selected, COLUMNLABELS are needed because there is no default column order for the MCMC_Prior_Groups input block.

MCMC_Prior_PDF Input Block (required)

The keywords for input block MCMC_Prior_PDF are as follows. "PDF" in the input block name stands for probability density function. Keywords MCLowerConstraint and MCUpperConstraint are required for an MCMC run.

As noted above, the keywords defined in the MCMC_Prior_Groups and MCMC_Prior_PDF input blocks may all be needed when MCMC=yes, but when MCMC_PREDICTION=yes, only the keyword ParamName defined in the MCMC_Prior_PDF block is used.

ParamName Parameter name (up to 12 characters; not case sensitive) – a character string that is used in a template file (template files are described in the UCODE_2005 documentation, Chapter 11). Each parameter name needs to be unique. If MCMC_Prediction=yes, the names listed here has to correspond to a parameter defined for the MCMC=yes, MCMC_Prediction=no run.

Naming convention for ParamName:

1) The first character has to be a letter of the set (A-Z, a-z); and

2) All remaining characters have to be a letter, digit, or member of the set:

_ . : & # @ (underscore, dot, colon, ampersand, number sign, at symbol).

The restrictions are needed for the parameter names to be used in the equations defined in Chapter 13 of the UCODE_2005 documentation.

GroupName Group name (up to 12 characters; not case sensitive). Each parameter needs to be a member of one group. **Default=DefaultPrior**.

The following two keywords define lower and upper constraints for the parameters. To improve the convergence of MCMC, out-of-bound parameter values in MCMC samples are folded into the bound; that is, they are set to values between the bounding values. The value, which we refer to as x, is determined as follows: (1) if x<min, then new_x=max-(min-x); (2) if x>max, then new_x=min+(x-max). "min" is MCLowerConstraint; "max" is MCUpperConstraint. This approach was suggested by J. Vrugt (University of California Irvine, 2012, written commun.).

Though MCLowerConstraint and MCUpperConstraint have default values, the defaults are very wide and generally are not useful. **Designation by the user is strongly advised.**

If the uniform distribution is used by default or designation using keyword PDFType, the constraints defined here are used as the bounds of the uniform prior distribution. In this case, MCMC generated values are constrained by the values specified using MCLowerConstraint and MCUpperConstraint. Any values designated for input variables PDFVar1 and PDFVar2 (see below) are ignored for parameters with a defined uniform prior distribution.

MCLowerConstraint- Smallest reasonable value for this parameter. Default=-(Huge real number). In absolute value, commonly about 10^{-38}.

MCUpperConstraint- Largest reasonable value for this parameter. Default=+(Huge real number). In absolute value, commonly about 10^{38}.

PDFtype The type of PDF. The eleven options are listed below. For additional information on these distributions, see Casella and Berger (2001). If multinormal is defined for parameters included in the definition of CovMatrix in the MCMC_Prior_Groups input block, the CovMatrix is used. Default=uniform.

PDFtype	PDFVar1	PDFVar2
uniform	lower bound	upper bound
normal	mean	std
log-normal	mean	std
beta	first parameter α[1]	second parameter β[1]
chi-square	degree of freedom *v*[1]	--
inv-chi-square	degree of freedom *v*[1]	--
scaled-inv-chi-square	degree of freedom *v*[1]	[1]scale σ^2
gamma	inverse scale β[1]	shape α[1]
inv-gamma	scale β[1]	shape α[1]
exponential	mean[1]	--
multinormal	mean	--[2]

[1] The value needs to exceed zero

[2] The variances and covariances are provided by CovMatrix in the MCMC_Prior_Groups input block.

PDFVar1 Value of the first variable used to define the prior PDF, as indicated above. Default=-(Huge real number). In absoluate value, commonly about 10^{38}.

PDFVar2 Value of the second variable used to define the prior PDF. For those distributions defined by only one variable, this second variable is not used but a dummy value (i.e., zero used in the following example) still needs to be entered here. Default=+(Huge real number). In absolute value, commonly about 10^{38}.

UCODE_2014 initiates Markov chains by drawing parameter samples from the prior distribution if UseRegResult=no and Restart=no in the MCMC_Controls block. Though PDFVar1 and PDFVar2 have default values for each distribution, their use is not recommended. It is strongly suggested that users define reasonable values for the distributions.

Example of the MCMC_Prior_Groups and MCMC_Prior_PDF input blocks when MCMC=yes for a problem with one independent, uniformly distributed parameter and five multi-normal distributed parameters.

```
BEGIN MCMC_Prior_Groups Table
Nrow=2  Ncol=3   COLUMNLABELS
GroupName       PDFType          COVMATRIX
PriGP1          uniform
PriGP2         multinormal        CovPrior
END MCMC_Prior_Groups
```

```
BEGIN MCMC_Prior_PDF TABLE
Nrow=6  Ncol=6   COLUMNLABELS
ParamName GroupName MCLowerConstraint MCUpperConstraint PDFVar1 PDFVar2
LAKERCH    PriGP2          1.0d-8          1.0d0     8.054E-04     0
KRB0       PriGP2          1.0d0           1.0d6     5.604E+03     0
RCH        PriGP2          1.0d-8          1.0d0     8.918E-04     0
KV         PriGP1          1.0d-8          1.0d0     1.0E-08    1.0E+00
K1         PriGP2          1.0d-3          1.0d3     4.458E+01     0
VANI1      PriGP2          1.0d-3          1.5d1     2.388E+00     0
END MCMC_Prior_PDF
```

```
BEGIN OBSERVATION_GROUPS TABLE
  nrow=   3   ncol=3  columnlabels
  GroupName       PlotSymbol    CovMatrix
         1          1           CovObsHead
         2          2           CovObsFlow
         3          3
END OBSERVATION_GROUPS
```

```
BEGIN MATRIX_FILES TABLE
  nrow= 3   ncol= 3  columnlabels
  MatrixFile              Nmatrices
  covmatrix.dat              1
  obsmatrix_head.dat         1
  obsmatrix_flow.dat         1
END MATRIX_FILES
```

File obsmatrix_flow.dat is listed below. See the UCODE_2005 documentation for formatting instructions and the file obsmatrix_head.dat distributed with UCODE_2014 for an example of compressed matrix format.

```
CompleteMatrix CovObsFlow
2 2
74529   0
0       4980038.56
```

Chapter 6: NEW RELATED PROGRAMS AND NEW AND CHANGED INPUT/OUTPUT PERFORMANCE

New and Changed Related Programs

New related computer programs and significant changes in existing programs used with UCODE_2005 or UCODE_2014 are listed briefly in Table 4. Some are also described briefly in the text. Please also see the listed references.

Table 3. New computer programs designed to work with files produced by UCODE and existing programs with significant changes.

Program	Documentation	Short description
MMA	Poeter and Hill (2007)	Multi-Model Analysis for evaluating alternative ways to simulate system features and properties.
OPR-PPR	Tonkin et al. (2007)	Calculates the Observation-Prediction (OPR) and Parameter-Prediction (PPR) statistics. OPR can be used to identify observations important to predictions. PPR can be used to identify parameters important to predictions. Additional statistics produced by OPR-PPR include the Observation-Parameter (OPA) statistic, among others. See comments in text about relation to PEST.
Sim_Adjust	Poeter and Hill (2008)	Identifies observations or predictions that have been omitted, or assigned a default value, by the process model, and provides user-defined alternate simulated values or defaults. A sequence of alternatives also can be defined to accommodate situations in which earlier alternatives also are not produced by the process model. For example if the value for head in a cell of a groundwater model is omitted or has an unreasonable value for use in computing residuals because the cell is dry, the user can use Sim_Adjust to provide instructions for using the head in the underlying cell and if that cell is also dry, then the head in the cell underlying that, and so on.
ModelMate	Banta (2011)	A Graphical User Interface to create UCODE input from MODFLOW2005 input files. To use ModelMate with other process models, template and instruction files need to be constructed separately. See text for an example ModelMate window.
GW_Chart	Hsieh and Winston 2002	A graphical post-processor that can graph results for many programs, including UCODE. Examples are provided in the text.
Model Viewer	Hsieh and Winston 2002	2- and 3-dimensional visualization of the sum of squared weighted residuals objective function. Examples are provided in the text.
Residual_ Analysis	Poeter et al. 2005	Supports residual analysis for results prior to conducting SVD. A brief description is provided in the text.

Program	Documentation	Short description
Concatonate_ IntConfPred	This report	Concatenates results from multiple runs to enable plotting of nonlinear confidence intervals. The capability is described in the text.

MMA

MMA (Multi-Model Analysis) uses output from multiple calibrated models. Results can be analyzed using a wide range of model discrimination criteria, including new user-defined criteria. AIC, AICc, BIC, and KIC are used by default. The analysis can also calculate posterior model probabilities and model-averaged confidence intervals on parameters and predictions. Prediction intervals also can be calculated. Several recent papers provide example of how the capabilities of MMA could be used and expanded.

Foglia et al. (2013) evaluates groundwater models that differ in the boundary conditions, geologic framework, and recharge and hydraulic conductivity distributions. The used cross-validation to test how well a set of model selection criteria could identify more accurate models. Results suggest some predictive power in criteria based on model fit and number of parameters (AIC, BIC, and so on), and difficulties for this non-linear model for criteria in which the sensitivities play a substantial rol (KIC).

Wellman et al. (2014) integrate more aspects of geology into the inverse modeling than has typically been the case. When using MMA, the approach of Wellman et al. means that more variable aspects of the system can be considered through the inverse modeling, and one might expect fewer alternative models. The tradeoff between what to include in the inversion, for which variables are continuous, and what to include in MMA-like analysis, for which variables are discrete, is an exciting topic of future research.

OPR-PPR

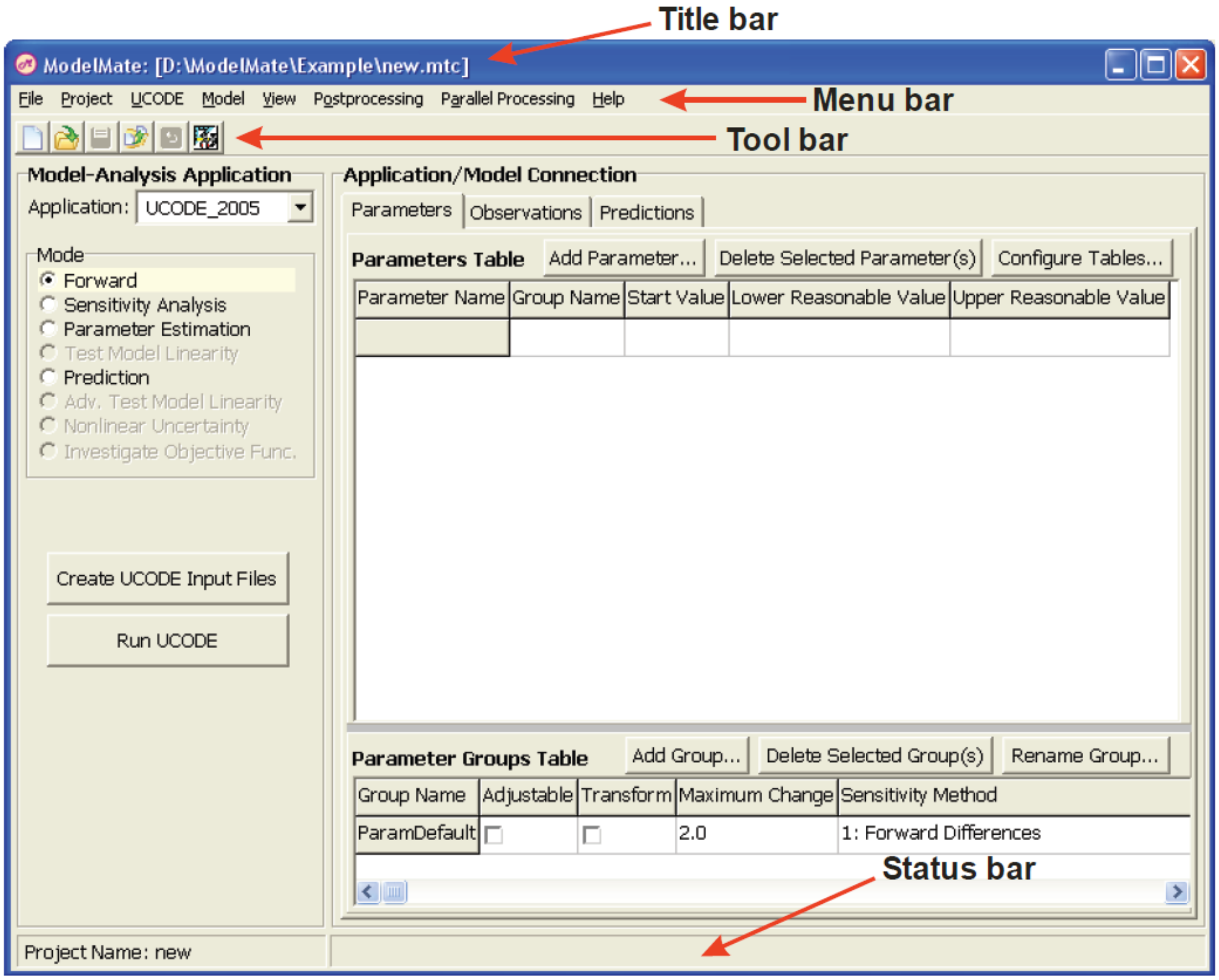
In OPR-PPR, results are directly comparable to PREDUNC and PREDVAR results produced by PEST, as discussed by Fienen et al (2010). Comparable statistics are the same except for two things:

(1) How percent change is calculated. OPR-PPR uses {100×(changed-base)/base} so that 0% means no change, negative values indicate a decrease, and positive values indicate an increase. PREDUNC and PREDVAR use 100× (changed/base) so that 100% means no change, values between 0.0 and just less than 100% indicate a decrease, and values larger than 100% indicate an increase.
(2) In PEST, the programming that calculates PREDUNC and PREDVAR use numerical methods that better support highly parameterized models than does UCODE.

ModelMate, A Graphical User Interface for Inverse Modeling

The Graphical User Interface ModelMate (Banta, 2011) can be used to organize parameters, obserevations, and predictions. It can create UCODE input from MODFLOW2005 input files. It can also be used with other process models, but then the template and instruction files would need to be constructed separately and information to define observations and parameters would need to be entered using ModelMate menus.

ModelMate can be obtained from the USGS software distribution site: http://water.usgs.gov/software/ModelMate/.



Application: UCODE_2005

Mode
- ● Forward
- ○ Sensitivity Analysis
- ○ Parameter Estimation
- ○ Test Model Linearity
- ○ Prediction
- ○ Adv. Test Model Linearity
- ○ Nonlinear Uncertainty
- ○ Investigate Objective Func.

Parameters Table

Parameter Name	Group Name	Start Value	Lower Reasonable Value	Upper Reasonable Value

Parameter Groups Table

Group Name	Adjustable	Transform	Maximum Change	Sensitivity Method
ParamDefault	□	□	2.0	1: Forward Differences

Project Name: new

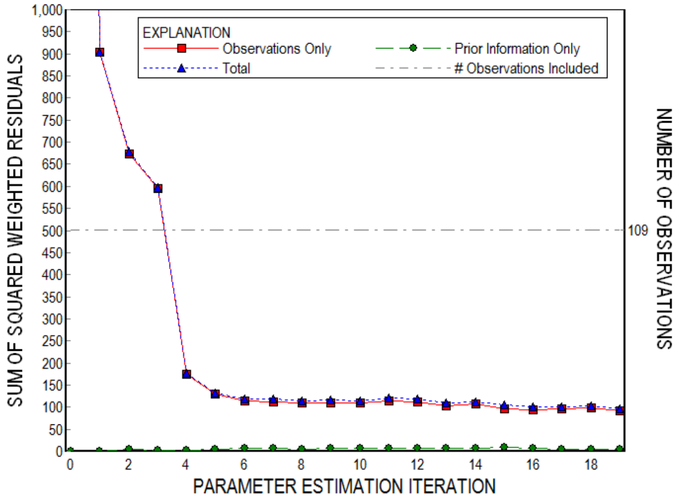
Figure 13. ModelMate window showing title bar, menu bar, tool bar, and Status bar. Note the tables for Parameters, Observations, and Predictions.

GW_Chart for Enhanced Evaluation of Results

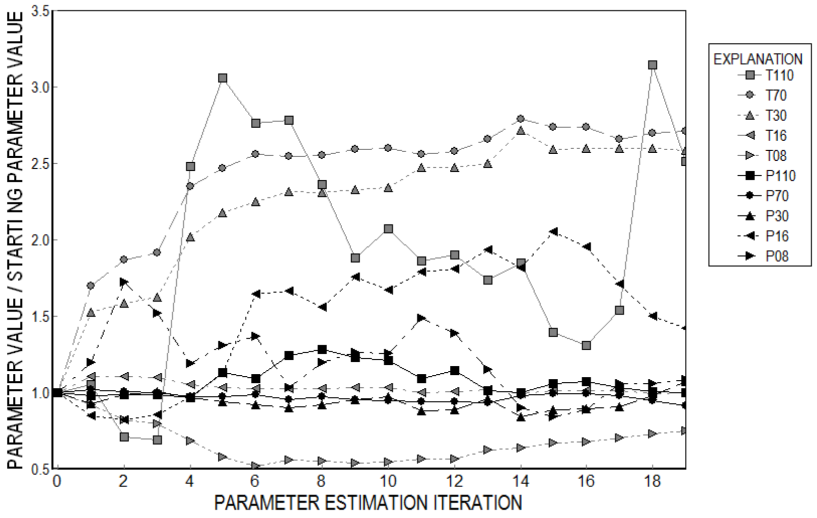
The contents of many of the data-exchange files produced by UCODE can be plotted using GW_Chart. Examples are shown in Figure 14. GW_Chart is very useful during the course of model calibration. GW_Chart may not produce the formatting desired for publication quality graphs, and data-exchange files are formatted to enable use of alternative plotting programs such as Excel.

GW_Chart can be obtained from the USGS software distribution site:
http://water.usgs.gov/nrp/gwsoftware/GW_Chart/GW_Chart.html

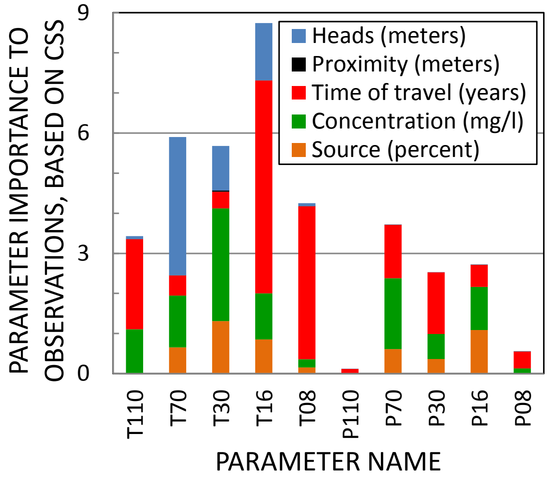
(A) Results from data-exchange file _ss, showing the progression of the sum of squared weighted residuals over the course of the parameter-estimation iterations. The number of observations is listed because observations can be omitted in some circumstances.



(B) Results from data-exchange file _pa, showing the progression of the parameter values over the course of the parameter-estimation iterations.



(C) Results from data-exchange files _scgrp, with bars of composite scaled sensitivities stacked to show the amount of information provided by five observation groups. This graph is discussed further in the section "Parameter and Observation Importance: A New Plot Using Data Exchange File _scgrp".



(D) Results from data-exchange file _pc, with estimated parameter values (link dots), calculated confidence intervals on the estimated values (black lines), and pre-calibration reasonable ranges on the parameter values (gray boxes).

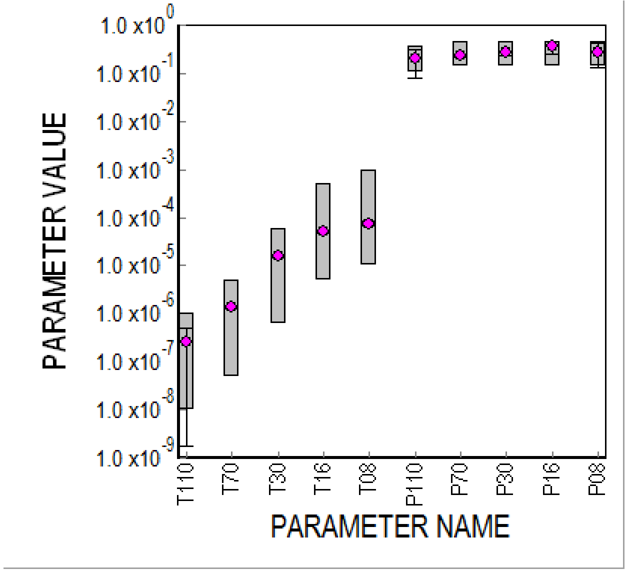
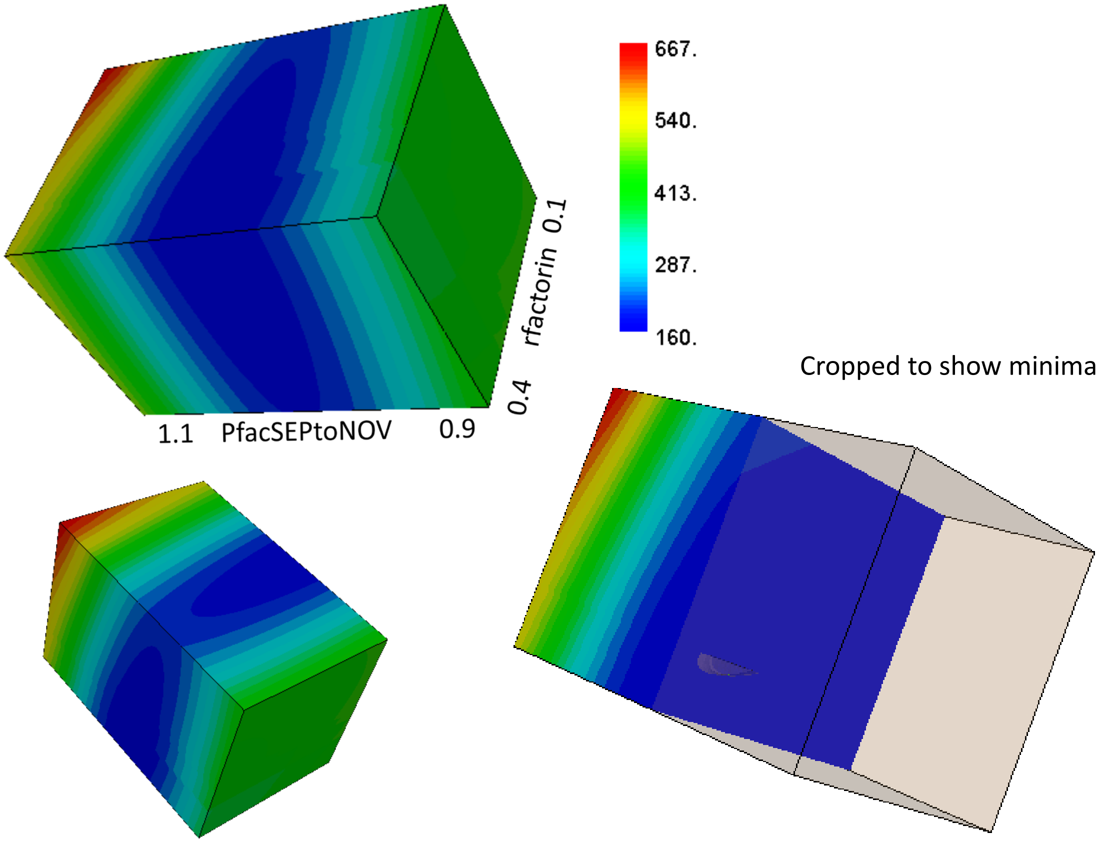


Figure 14. Example graphs produced by GW_Chart. These graphs are from the test problem described in Hanson et al. (2013).

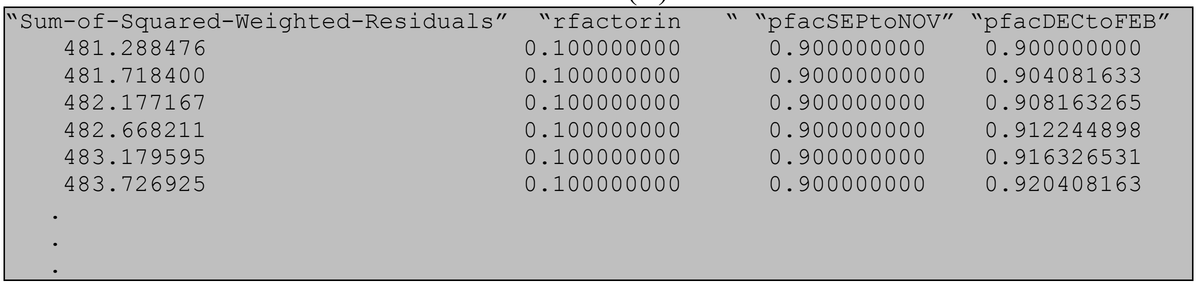
Model Viewer for Plotting Objective Function Surfaces

Model Viewer (Hsieh and Winston, 2002), starting with version 1.2, can read a data-exchange file with filename suffix _sos and columns composed of the objective function value followed by values of two or three parameters. Results for three parameters from a Water Balance Model (McCabe and Markstrom, 2007) and part of the associated _sos file are shown in Figure 15.

(A)



(B)



```
“Sum-of-Squared-Weighted-Residuals”  “rfactorin   “ “pfacSEPtoNOV” “pfacDECtoFEB”
    481.288476                      0.100000000      0.900000000    0.900000000
    481.718400                      0.100000000      0.900000000    0.904081633
    482.177167                      0.100000000      0.900000000    0.908163265
    482.668211                      0.100000000      0.900000000    0.912244898
    483.179595                      0.100000000      0.900000000    0.916326531
    483.726925                      0.100000000      0.900000000    0.920408163
  .
  .
  .
```

Figure 15. (A) Three-dimensional plots of an objective function. The figure in the lower right is cut away to show the region around the minimum value of the objective function, which appears in grey. (B) First 6 of 125,000 lines of the UCODE output file with filename extension _sos which was used to create the figures shown. The results are from the Water Balance Model (WBM) of McCabe and Markstrom (2007). The discontinuity stems from defining an inadequate burn-in period. The figures were produced using Model Viewer (Hsieh and Winston, 2002)

Residual_Analysis

Residual_Analysis, has been modified to allow for analysis of residuals from data-exchange files generated before implementing the SVD parameter estimation in addition to analyzing residuals from the final optimized values. The code is back compatible. To use the new feature, provide a second argument “presvd” on the Residual_Analysis command line. Given that second argument, Residual_Analysis uses data exchange files with the additional _presvd extension. These files include the following.

fn._dm_presvd,
fn._mv_presvd,
fn._paopt_presvd,
fn._su_presvd,
fn._supri_presvd,
fn._w_presvd,
fn._wt_presvd, and
fn._wtpri_presvd.

where fn is replaced by the first argument on the Residual_Analysis command line

The output from Resudial_Analysis is stored in files with the following extensions

fn._rd_presvd,
fn._rg_presvd,
fn._rc_presvd,
fn._rb_presvd, and
fn.#resan_presvd.

If the optional input file, fn.rs, is used, it needs to have the additional file extension, and thus would be fn.rs_presvd.

CONCATENATE_INTCONFPRED

CONCATENATE_INTCONFPRED is a program to concatenate multiple nonlinear confidence interval output files for post-processing. These files are produced when using the nonlinear intervals mode of UCODE (Poeter et al., 2005, Chapter 17).

When calculating nonlinear uncertainty intervals, different runs of UCODE are often used to calculate different nonlinear interval limits because of lengthy runs and convergence problems. Each limit of each interval requires a regression, and each of these regression runs is totally independent. For example, if uncertainty intervals are being calculated for five predictions, 10 regressions are needed. These can be calculated using 10 different runs of UCODE, each on a separate computer. CONCATENATE_INTCONFPRED helps accumulate these intervals into one file for easier polotting by GW_Chart or other software.

To facilitate these different runs, new files with date/time stamps in the filename are produced for files with extensions
_intconf,
_intconfpar,
_intpred, and
_intpredpar.
The new file names are, for example, fn–2008_06_21–22.17._intconf.

For each UCODE nonlinear uncertainty run, files

fn._intconf,
fn._intconfpar,
fn._intpred, and
fn._intpredpar

are produced. As before, they contain the intervals calculated in the most recent run. To obtain a full list of interval limits, values from the time-stamped files can be moved into these files without the time stamps. Once this is accomplished, the nonlinear intervals can be plotted in graph form using GW_Chart.

The new post-processing code Concatenate_IntConfPred can be used to merge output from the UCODE nonlinear interval mode. After executing UCODE in the non-linear interval mode, a number of files are produced. The files of interest in this section are the time-stamped copies of the fn._intconf and fn._intpred files. These names could be, for example,

fn –2013_03_08–14.34._intconf

or

fn –2013_03_08–16.25._intpred.

Time-stamped output is created by the non-linear interval executions of UCODE because the non-linear interval runs can be very long. Creating time-stamped copies of the most valuable output files prevents inadvertent overwriting of the results. These files are readily viewed in GW_Chart. It is preferable to view all of the intervals of interest in one graph. However, typically numerous executions of the UCODE nonlinear interval mode are needed before all of the acceptable intervals are calculated. The code described in this section (Concatenate_IntConfPred) facilitates combining those time-stamped files into one file with a name that includes “concatenated” for easy identification, such as fn._concatenated_intconf or fn._concatenated_intpred.

Numerous executions of the UCODE nonlinear interval mode are typically needed for various reasons. For example, because the runs are expected to be long, only a few limits might be included the first run in order to assess the situation and get a better feel for appropriate settings. Let’s call that time-stamped file A. Some of the limits in that file may have converged to within an acceptable deviation from the goal, but others may not have converged, or may not be within an acceptable percentage deviation from the goal. Consequently another execution of the UCODE nonlinear interval mode might be undertaken with different criteria (e.g., more iterations using MaxIter and/or smaller tolerances for TolIntP, TolIntS, and/or TolIntY). Let’s call the resulting time-stamped file B. If most of those interval calculations converged within a reasonable percentage deviation of the goal, then yet another execution of the UCODE nonlinear interval mode might be undertaken with additional items (i.e., parameters and/or predictions) and if both the upper and lower limits were not included previously, the limits that were not previously calculated might be included in this run. We will call the resulting time-stamped file C. Perhaps many more UCODE nonlinear interval mode executions will be made before the user is satisfied with all of the results. In addition to the variations described in this paragraph, the user may at times execute the UCODE nonlinear interval mode for confidence versus prediction limits and/or for individual versus simultaneous intervals for the same items. Note that these choices are made for an entire UCODE nonlinear interval mode execution. Thus an entire time-stamped file will contain either confidence or prediction limits (but not both) and either individual or simultaneous intervals (but not both). It is not valid to mix confidence with prediction limits, or individual with simultaneous intervals.

Once the user is satisfied with the set of nonlinear limits that are available in all of the time-stamped files, it is desirable to have all of the confidence limits in one _intconf file and/or all prediction limits in one _intpred file. That is when the Concatenate_IntConfPred program is executed. The simple input file described below contains 1) the indication of whether confidence or prediction limits are of interest and 2) the root names of the time-stamped files to be merged.

A number of issues may be encountered when merging the files. The potential issues and the action Concatenate_IntConfPred takes to resolve them are listed in the following Table. The code runs quickly so the user can identify the issues and address them in a short time. It is recommended that the user make a copy of all the time-stamped files in a separate folder because some of the solutions involve the user editing the time-stamped files. Maintaining copies elsewhere will prevent an inadvertent error that may cause loss of time while waiting for a UCODE nonlinear interval mode to run again.

Execution

The CONCATENATE_INTCONFPRED run command is of the form

```
path:\ CONCATENATE_INTCONFPRED.exe  fnconcat
```

where

path:\ is the relative or absolute path to the CONCATENATE_INTCONFPRED.exe on your computer (alternatively you could specify this in your system path variable)

fnconcat is the filename prefix for the CONCATENATE_INTCONFPRED input and output files, which may, but does not need to be the same as that of the previous executions of UCODE (spaces are not allowed in fn, even on operating systems that allow spaces in filenames)

Input Files for CONCATENATE_INTCONFPRED

CONCATENATE_INTCONFPRED relies on previously generated data-exchange files that are file-stamped copies of the fn._intconf and fn._intpred files. These names are for example, fn–2013_03_08–14.34._intconf or fn–2013_03_08–16.25._intpred.

A user-prepared input file named fnconcat.concatenate is needed to execute CONCATENATE_INTCONFPRED, where fnconcat is the root filename defined on the command line. It does not need to match the root filename of the earlier UCODE runs. The first line of this file includes an indicator of whether confidence limits or prediction limits will be merged (either conf or pred, which are not case sensitive). Subsequent lines list the root names of the files to be merged.

Example of input the CONCATENATE_INTCONFPRED program:

```
pred
ex1-2013_03_08-23.17
ex1-2013_03_08-14.34
```

Output Files for CORFAC_PLUS

Each run of CONCATENATE_INTCONFPRED produces two output files: the main output file with file extension #concatenated._int* and a data-exchange file with file extension _concatenated._int*. The * is replaced by 'conf' or 'pred' depending on the word on the first line of the input file, fnconcat.concatenate. The * is replaced by 'conf' when the first line reads conf and 'pred' when it reads pred.

The CONCATENATE_INTCONFPRED main output file, with file extension #concatenated._int* echoes the information as it is read from the list of time-stamped files and reports errors with suggestions for correcting the situation.

Table 4. Potential problems with Concatenate_IntConfPred files and suggested actions.

Potential Problem	Concatenate_IntConfPred Action and Suggested User Action
A listed time-stamped file is not found.	After checking for the existence of all the listed time-stamped files, the program terminates and lists the files that did not exist.
The same item and limit is repeated in the time-stamped files, but all associated information is identical.	This means that calculations for the limit were repeated for more than one run of UCODE. The item is listed in the main Concatenate_IntConfPred output file (file name suffix starts with #), the situation is ignored and the program continues.
The same item (name and limit, i.e. upper or lower) is repeated and any aspect of associated information differs. For example, this will occur in the common situation in which UCODE has been run in the nonlinear interval mode for the same item and limit but with more iterations or a different closure tolerance.	After reading all files and items, an error message is written to both output files along with a list of all items with identical name and limit (i.e., upper or lower), but different associated information and the associated time-stamped files. The program terminates. The user is advised to either revise the list of files provided to Concatenate_IntConfPred in the input or remove the unwanted limits from the time-stamped files.
The first line of the input file is 'conf' and the header of a time-stamped file reads PREDICTION LIMT, or the first line of input is 'pred' and the header of a time-stamped file reads CONFIDENCE LIMT.	After reading all files in the list, an error message is written, offending files are listed in both output files and the program terminates. The user should remove the offending files from the list of files in the Concatenate_IntConfPred input file.
All limits are not of same type (for example, limits for individual and simultaneous intervals are reported).	After checking all files, an error message is written to the UCODE main output file and the associated time-stamped files. The program terminates. The user can determine the offending files as those in the Concatenate_IntConfPred main output file (#) that are followed by the message "!!!! MIXED INDIVIDUAL and SIMULTANEOUS LIMITS !!!!". The user is advised to either revise the list of files provided to Concatenate_IntConfPred in the input or remove the unwanted limits from the time-stamped files.
A lower and upper limit is included for the same item name but the upper limit is lower than the lower limit.	All files and items are read. The item name and file name of each occurrence of LowerLimit > UpperLimit is written to the main (#) Concatenate_IntConfPred output file. The program terminates. The user is advised to either rerun those intervals or remove them from the time-stamped files.

CONVERT_PEST_PRIOR_INFO

CONVERT_PEST_PRIOR_INFO is a program that reads a pest file, finds the prior items and uses them to create a UCODE Linear_Prior_Information input block. This allows the PEST capabilities to create prior information for parameters defined at pilot points to be used in constructing UCODE input files. For models with more than about a hundred parameters, the support for highly parameterized models provided by PEST is likely to reduce execution time relative to UCODE. On the other hand, UCODE has sensitivity analysis statistics that might be useful in some circumstances.

The command line needed to run the program CONVERT_PEST_PRIOR_INFO is

```
convert_pest_prior_info.exe    name-of-pest-file    base-name-of prior-items-in-pest-file
```

The two arguments are as follows. A set of input and output files are provided with the UCODE distribution in a directory called convert_pest_prior_info.

Name-of-pest-file — Full name of the PEST main input file. This is also used to create the name of the files with the UCODE Linear_Prior_Information input block. The new file name is created by appending ".out" onto the input specified for name-of-pest-file.

In the example distributed with UCODE the name of the PEST main input file is "06-upestrun-p.pst" and the file containing the UCODE Linear_Prior_Information input block is named 06-upestrun-p.pst.out.

base-name-of prior-items-in-pest-file — Base name for prior items to be written using the program convert_pest_prior_info.

The program only converts prior information for which the base name matches this keyword. In the example distributed with UCODE the base name is "r".

A short summary of what was done is in the output file ConvertPestPriorInfo.#out.

Changed UCODE Input/Output Performance

This section describes changes in the UCODE data-exchange files, and changes to the input blocks that make up the main input file that are not described in the previous sections. The latter includes changed functionality of some keywords and some new keywords. Two input blocks are involved in the changes. All changes are minor and will affect few users. Some of what is documented here was implemented in versions of UCODE_2005. Version numbers for these changes can be found in the history file of the UCODE_2005 distribution.

The changes are implemented such that UCODE_2014 is fully backward compatible.

Modified Data-Exchange Files

There is one new data-exchange file and changes to two others.

Parameter and Observation Importance: A New Plot Using Data Exchange File _scgrp

The _scgrp data exchange file was changed so that stacked graphs of CSS, such as that shown in Figure 16 could be plotted. The change is simply adding the number of observations in each group. This allows a more useful graphical display of how observations contribute to the sensitivities calculated for each parameter than the original display supported by previous versions of MODFLOWP (Hill, 1992), MODFLOW-2000 (Hill et al, 2000), and UCODE_2005, and used by Barlebo et al. (1998). Like any graph of composite scaled sensitivity, the results are for individual parameters and do not display the consequences of parameter correlation. The effects of parameter correlation can be evaluated using other statistics such as leverage, Cook's D, and DFBETAS.

Figure 16 shows how the new method is used to display the contributions of five types of observations to the estimation of 10 parameters. In this sample problem, particle tracking is used to produce simulated equivalents for four types of field observations typically derived from concentration measurements and provided through the new MODPATH-OBS program (Hanson et al., 2013):

1. direction of transport (called proximity observations in the figure),
2. travel time (only available once the basic direction of transport is simulated correctly),
3. "concentrations", used in this work to approximate PCE concentrations, but often used to simulate water age, and
4. source of water, which generally is supported by data identifying the source areas of recharging water.

The stacking shown in Figure 16 is a useful way to depict observation contribution to parameter sensitivity. The calculations and data-exchange files behind the stacked graphs are described in Appendix F. The units of the observations are listed in the figure; the observations are normalized as a step in producing the graph.

Data-Exchange File fn._eig

The _eig data-exchange file is produced when keyword EigenValues is yes in the UCODE_Control_Data input block. Data-exchange file fn._eig contains eigenvalues and eigenvectors of the parameter posterior variance-covariance matrix (which is in the _mv data-exchange file) scaled by the parameter values. Small eigenvalues are associated with more important eigenvectors; large eigenvalues are associated with less important eigenvectors. More sensitive parameters tend to dominate more important eigenvectors.

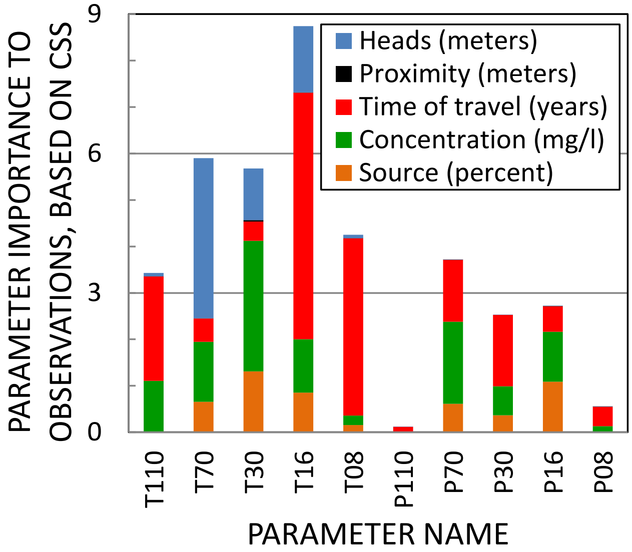


Figure 16. Figure produced using the _scgrp data-exchange file. This figure was produced using GW_Chart (Winston, 2000)

The eigenvectors are the same as the singular vectors produced in SVD; the eigenvalues equal the reciprocal of the square of the singular values.

For UCODE_2014, the format of the fn._eig file has been changed to facilitate user reading of the file. Although data-exchange files are primarily intended for computer processing, users often view the files and the new format makes it easier to see the association between the eigenvalues and their associated vector elements. The file is presently formatted as shown in Figure 17. Previously the eigenvalues were listed as the first column of the file, making it less clear that each eigenvalue is associated with one eigenvector.

```
"Variance-Covariance Matrix of the Parameters scaled by parameter
values: Eigenvalues and Eigenvectors"
                                          "EIGENVALUE            1"
 "EIGENVALUE FOR EACH VECTOR  "           0.2165814084099306E-01
 "PARAMETER FOR EACH VECTOR ELEMENT "     "EIGENVECTOR
 HK_1                                     -0.7148319210731826
 K_RB                                     -0.2397959977006962E-02
 VANI                                      0.7405157406540819E-16
 VK_CB                                    -0.2452155814601121E-02
 HK_2                                     -0.1241457555251904
 RCH_1                                     0.1794793381884821
 RCH_2                                     0.6643632740324659
```

Figure 17. Part of an example _eig file, including the header line (here shown on two lines) and the first two columns.

Time Stamps for Nonlinear Uncertainty Output

When calculating nonlinear uncertainty intervals, different runs of UCODE are often used to calculate different intervals limits because of lengthy runs and convergence problems. To facilitate these different runs, new files with date/time stamps in the filename are produced for files with extensions _intconf, _intconfpar, _intpred, and _intpredpar. The new file names are, for example, fn–2008_06_21–22.17._intconf. For each UCODE nonlinear uncertainty run, files with extensions _intconf, _intconfpar, _intpred, and _intpredpar are still produced. As before, they contain the intervals calculated in the most recent run. To obtain a full list of interval limits, values from the time-stamped files may need to be moved into these files without the time stamps. Then the nonlinear intervals can be plotted using GW_Chart. The new program Concatenate_IntConfPred allows this to be done as part of a batch file, thus reducing the chance of error.

Modified Keywords and Input Blocks

Changes have been made to the UCODE_Control_Data and Reg_GN_Control input blocks.

UCODE_Control_Data Input Block

DataExchange. The keyword **DataExchange is now always set to Yes**. A designation of no is ignored.

New Keyword: Reactivate. Reactivate governs performance of UCODE when the regression converges, or the regression fails to converge and stats_on_nonconverge=yes. The original UCODE_2005 functionality is described below under Reactivate=no.

Reactivate Final, Starting, and No. **Default=Final**.

Final: all parameters (including omitted parameters) are set to the values associated with the iteration having the lowest sum-of-squared-weighted-residuals. The final sensitivities and statistics are generated for the parameters that were defined as adjustable at the start of the regression.

Starting: as for Reactivate=final except that omitted parameters are set to their starting values. This method has the advantage of using values for omitted parameters that are more likely to be in the middle of their expected range. However, the use of a mixed set of parameter values (that is, starting values for the omitted parameters and altered values for the other parameters) generally results in a model fit that is worse than the lowest sum-of squared-weighted residuals achieved during regression. It is advisable to repeat the regression with the omitted parameters set to adjustable=no. To bring this to the attention of the user UCODE prints warning messages if the resulting sum-of-squared-weighted-residuals is larger than the lowest sum-of-squared-weighted-residuals identified during the regression.

No: all parameters (including parameters omitted at the end of the regression run) are set to the values associated with the iteration having the lowest sum-of-squared-weighted-residuals. The final sensitivities and statistics are generated only for the parameters that were active for the iteration with the lowest sum-of-squared-weighted-residuals.

Reg_GN_Controls Input Block

New Keywords: Scaling, OmitWeight.

New keyword Scaling influences the trust region regression algorithm accessed using keyword TrustRegion, which is described in the UCODE_2005 documentation (Poeter et al., 2007, p. 63). Scaling=no adds the option to not scale the least squares matrix. Scaling of the matrix is described by Hill and Tiedeman (2007, p. 72). Not scaling the matrix may improve regression performance when the regression is not able to reduce the objective function value (Dennis Gay, and Welch, 1981 and our own experience).

Scaling	Yes, No. **Default = Yes**	
	Yes:	scaling is used.
	No :	scaling is not used.

A new keyword OmitWeight allows the user to control how observations are omitted. The omission is accomplished by reducing the value of the weight: the weight is made so small that the observation does not affect the regression. As discussed in the UCODE_2005 documentation (Poeter et al., 2005, p. 62), the use of keyword OmitDefault to define numbers which, when encountered as a simulated value, indicate that the observation is to be omitted from the regression. The program Sim_Adjust can be used to define alternative simulated values; for example, in groundwater models if the observation cell goes dry, Sim_Adjust can be used to tell UCODE to use the head in the underlying cell. If no alternative values are defined or none can be simulated, the situation can be controlled using OmitDefault.

Previously, the weight was reduced without the possibility of user intervention, but situations have occurred when the defined procedure did not produce a sufficiently small weight. The default procedure has now been changed to be more robust. In addition, the user has been provided the ability to control the applied weight using the new keyword OmitWeight.

OmitWeight the weight used for omitted observations. **Default** = 1E-70

Multiplying OmitWeight times the OmitValue needs to produce a value that is small relative to most the other weighted residuals.

The OmitValue functionality is intended to allow the regression to continue through parameter sets for which a few observations are not be calculated. If the final estimated parameter values result in a simulation that is unable to produce simulated values for any observations, inspection of those observed and simulated values is advised.

For UCODE calculation of some statistics, such as the standard error of the regression, AIC, BIC, KIC, linear and nonlinear confidence intervals, and so on, the value of observations used (often called N or ND in the equations) will include the omitted observations. Elimination of these observations from the data set by including them in an observation group for which UseFlag = no or removing them from the data set is required to obtain correct calculation of the affected statistics. This generally requires an additional sensitivity analysis mode run, but is not expected to occur frequently.

Chapter 7: USING A MATLAB MODEL WITH UCODE, by_Marc J. Rubin[1] and Allan K. Haas[2]

Many scientists and engineers use MATLAB to perform a variety of analyses, and as such there is likely to be a substantial reservoir of MATLAB code that could be useful to UCODE users. In this chapter, we show that it is rather straightforward to combine these powerful tools by running a MATLAB model from UCODE. For this application, MATLAB is run without its usual GUI. UCODE becomes the controlling program and makes calls to run MATLAB. UCODE calls MATLAB multiple times, adjusting the values of the MATLAB model input parameters for each MATLAB call based on the designations in the UCODE main input file and a constructed UCODE template file. Results are read from MATLAB text-file output using a UCODE instruction file.

When running with UCODE, the MATLAB GUI is not present but the executed MATLAB code could still produce graphical displays and other useful results. It may be desirable for the user to employ MATLAB generated graphics and other displays to confirm the MATLAB model integrity and provide additional input. Using MATLAB graphics will slow down the execution. Therefore, it is recommended that MATLAB run graphics free during analyses requiring multiple forward model calls conducted by UCODE.

The process makes use of the command line options and execution that MATLAB provides. Briefly, the MATLAB execution command is placed in a Microsoft batch file, which UCODE then calls to run the MATLAB model. The MATLAB model is configured to work with the input and output file structure that UCODE requires; that is, input and output files need to be ASCII files and associated UCODE template and instruction files need to be constructed and the filenames listed in the UCODE main input file Model_Input_Files and Model_Output_Files input blocks. The combination of these files construct a very simple, flexible, and robust communication interface between the two main software facilities, allowing operational arguments and data to be exchanged between them. Finally, the MATLAB model needs to call MATLAB's "exit;" command when the model completes. This causes MATLAB to terminate and return system control to UCODE.

Two preparatory steps are needed before MATLAB can be used with UCODE. First, as stated above, existing or new MATLAB codes must be modified or constructed to conform to UCODE file interchange requirements. Second, the MATLAB-UCODE interface files must be configured. After these file preparations, the three steps outlined below are needed to operate MATLAB from UCODE without the MATLAB graphical user interface (GUI). The following presents the statements and arguments that are available to run MATLAB (without its typical GUI), from

[1] Oregon State University Cascades, School of Electrical Engineering and Computer Science, Bend, Oregon, USA marc.rubin@osucascades.edu
[2] Colorado School of Mines Department of Geophysics; now at HydroGEOPHYSICS, Inc., Golden, Colorado, USA mygeophysics@gmail.com

UCODE. Step one also provides an example using iTOUGH2 instead of UCODE to give users another example of how MATLAB programs can be used with inverse modeling software..

<u>Step One</u>: Running a MATLAB Model from the Command Line

Before discussing how to run a MATLAB model from UCODE, one must first understand how to run a MATLAB model from the command line. From a Window's command line, the following command will run a MATLAB script named "model" (i.e., model.m):

CMD> MATLAB –r model

where ***MATLAB*** is the full executable path of MATLAB and ***model*** is a MATLAB script (i.e., "model.m") that is within MATLAB's path. The ***–r*** command line option forces MATLAB to run the command or script that follows. To make MATLAB run faster, other command line options are added:

CMD> MATLAB –nosplash –nodesktop –r model

The ***–nosplash*** command line option disables MATLAB's startup splash screen and the ***–nodesktop*** option disables MATLAB's Java based graphical user interface (GUI).

The command used by iTOUGH2 to run MATLAB is as follows, where what is on two lines here would appear on one line (S. Finsterle, written commun. 2014)

"matlab -nodesktop -nosplash -nojava -wait -automation –r"

"run('c:\path\to\myMATLABCode.m'); exit"

<u>Step Two</u>: Running a MATLAB Execution Command from a Microsoft Batch File

The command needed to execute the MATLAB model is specified in the run.bat file which contains the following line.

CALL MATLAB –nosplash –nodesktop –wait –r model

MATLAB's ***–wait*** command line option forces the batch script to wait until the MATLAB process terminates, which is crucial for MATLAB to work with UCODE. To reiterate, the MATLAB model MUST abide by UCODE's input and output file requirements. Additionally, the MATLAB model MUST call MATLAB's "exit;" command to allow the MATLAB process to terminate.

<u>Step Three</u>: Modify *UCODE_main.in*

In the Model_Command_Lines input block of the UCODE main input file, call a Microsoft batch script named "**run.bat**", which contains the command to execute a MATLAB model from the command line. The line in the input block will look like this:

Command = 'run.bat'

Chapter 8: LINKING UCODE WITH A PROGRAM THAT IS RUN THROUGH A GRAPHICAL USER INTERFACE, by Assaf Wunsch[1]

As an example of a program run through a graphical user interface (GUI) in a MS Windows environment, this section focuses on HYDRUS.

The finite-element software HYDRUS (from PC-Progress) has become a popular tool for simulating saturated and unsaturated flow in the subsurface. A 1-dimensional (1D) calculator (Šimůnek et al., 2005) is available for free; the 2D/3D version, which enables complex domain geometries (Šimůnek et al., 1999), is not. HYDRUS has a built-in inverse simulation feature, both in the free and the paid versions. However, only parameters related to flow (hydraulic conductivity, van-Genuchten parameters, dispersivity, etc.) can be optimized. In addition, the user has little control over the settings of the optimization calculation itself, such as convergence criteria and incremental change of each parameter during iterations. This guide is therefore intended for users of HYDRUS who wish to take advantage of some of the complex features of UCODE, or optimize parameters in HYDRUS that are not directly related to the intrinsic flow properties of the porous media (for example, boundary conditions and reaction rates).

The basic steps are as follows.

a) Construct a model as usual using the HYDRUS GUI.
b) Proceed as described below to do the same HYDRUS run using a batch file and create the needed input files. Ensure that the batch file process is producing the same simulation results as obtained when using the GUI.
c) For UCODE, use the batch file and construct the needed template and instructions files.
d) Run HYDRUS using UCODE.

The following sections provide specific instructions for accomplishing these steps.

Calling HYDRUS from UCODE

One of the appeals of HYDRUS is its GUI. However, running HYDRUS simulations from UCODE requires executing HYDRUS from the command prompt, or through a batch file. Here the batch

[1] Colorado School of Mines Department of Civil and Environmental Engineering. Now at Leppert Associates, Golden, Colorado, USA awunsch@leppertassociates.com

files for running HYDRUS are described. The location and name of the batch files should be defined in the “**MODEL_COMMAND_LINES**” block of the main UCODE input file.

HYDRUS 1D

An example of a batch file for running HYDRUS 1D on a MS Windows operating system is given below:

------------ run_hydrus_1D.bat ------------

```
cd\
cd program files
cd pc-progress
cd hydrus-1d 4.xx
h1d_calc C:\Hydrus_simulations\1d\col\my_simulation
```

------------ end file ---------------------------

In this example, the first four lines simply point to the directory that contains the HYDRUS_1D executable, and this can be done in a different way. Notice the syntax of the last line in the batch file: “**h1d_calc**” refers to the HYDRUS_1D executable, “**h1d_calc.exe**”; the term “**my_simulation**” refers to the HYDRUS project file “**my_simulation.h1d**”. Generation of the project file is discussed below.

HYDRUS 2D/3D

An example of a batch file for running HYDRUS 2D/3D is given below:

------------ run_hydrus_2D.bat ------------

```
cd\
cd program files
cd pc-progress
cd hydrus 1.xx
h2d_calc.exe
```

------------ end file ---------------------------

As with the previous example, the first four lines simply redirect the path to the location of the HYDRUS 2D executable, and the last line contains the executable itself. However, notice that the project name itself is missing. To instruct HYDRUS 2D to run a specific project, there needs to be a file called “**level_01.dir**” in the directory which contains the HYDRUS 2D executable (for the example above, the pathname of the file would be “**c:\program files\pc-progress\hydrus 1.xx\level_01.dir**”). The file “**level_01.dir**” needs to contain the project directory. For example, it might be:

------------- “level_01.dir” ---------------

```
c:\hydrus_simulations\2d\H3D_My_2D_Simulation
```

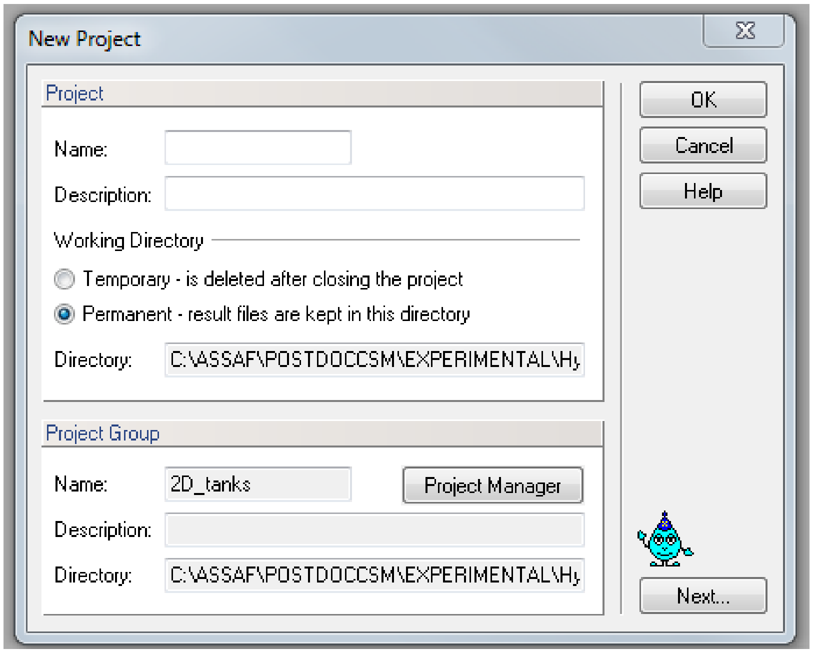
------------ end file ----------------------

where “**My_2D_Simulation.H3D**” is the name of your project. Unlike in the HYDRUS 1D example, here the **directory** where all the project files are kept is listed, **not the main project file** (i.e., not the directory that contains the actual “**.H3D**” file).

Important note: Do not forget to change the content of "**level_01.dir**" – i.e., the location of the HYDRUS project - when switching from one project to another. Otherwise the same project will be optimized again and again. When you are done with the optimization, the file “**level_01.dir**” can be deleted.

HYDRUS Input Files

By default, HYDRUS collapses and compresses all the text and non-text input and output files when a project is closed. To keep the project files in a directory where you can view them when the project is closed, you need to change some settings in HYDRUS. When starting a new project, the following menu appears, requiring the user to assign a project name, description etc. Under “Working Directory” choose “Permanent” instead of “Temporary”. That will keep the project files accessible on your hard drive.



New Project

Project

Name:

Description:

Working Directory

○ Temporary - is deleted after closing the project

● Permanent - result files are kept in this directory

Directory: C:\ASSAF\POSTDOCCSM\EXPERIMENTAL\Hy

Project Group

Name: 2D_tanks

Description:

Directory: C:\ASSAF\POSTDOCCSM\EXPERIMENTAL\Hy

Most of the HYDRUS input parameters are located in a file called “**Selector.in**”, both in the 1D and the 2D/3D versions. This file should be used to create a template file as described for the UCODE “**MODEL_INPUT_FILES**” input block.

HYDRUS Output Files

HYDRUS produces several output files, some are text files (ASCII) and some are not. The text files can be used to provide UCODE with simulation results. In the UCODE input file, HYDRUS text output files can be listed in the “**MODEL_OUTPUT_FILES**” input block. HYDRUS output text files commonly of interest include, for example, the cumulative fluxes across boundaries (“**cum_Q.out**”) or mean velocities (“**v_Mean.out**”). In HYDRUS 1D, most of the output data is stored in “**T_Level.out**”. However, the most useful output file is probably the output file for the observation nodes that are “planted” in the model, as discussed in the following sections.

HYDRUS 1D Observation Node Output

Data recorded for the observation nodes is kept in the output file “**Obs_node.out**”. Observation nodes are identified in “**Obs_node.out**” by their location in the 1D model. Node numbers in HYDRUS 1D start with “1” at the top model boundary, and increase linearly until the bottom model boundary is reached. An observation point that is located on node 29 will simply be called “**Node (29)**” in the output file.

HYDRUS 2D/3D Observation Node Output

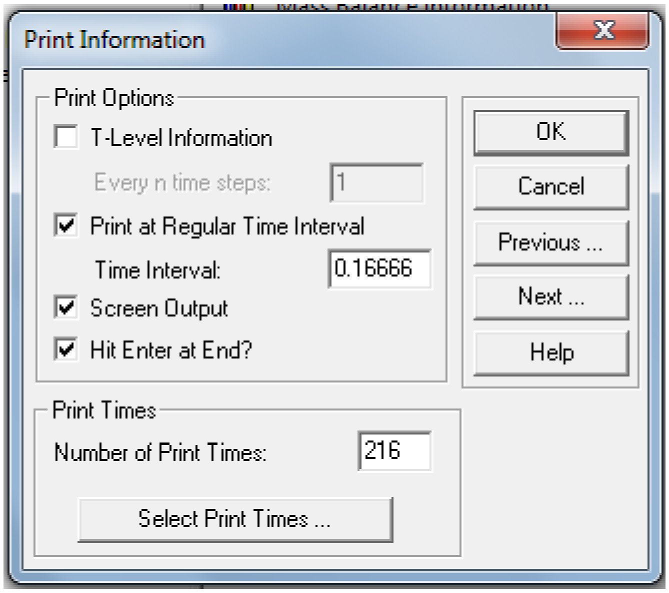
Data recorded for selected observation nodes is kept in the output file “**Obsnod.out**” (notice the omission of the underscore and the letter “e” in the file name relative to the equivalent HYDRUS 1D output file). As in the 1D version, observation nodes are identified by the node number. However, the assignment of node numbers in a finite-element mesh is complex, and does not follow rules such as incremental increase from domain top to bottom, left to right etc. Therefore, the user should make sure to identify the node number using the HYDRUS 2D/3D GUI.

Instruction Files for UCODE

Instruction files contain output-specific instructions, enabling UCODE to obtain simulated values that are then compared to observed values. The instruction files for HYDRUS can be constructed like any other instruction files described in the UCODE manual. A crucial aspect for matching the correct observation to the correct model output is consistency in the output files. However, the default in HYDRUS is to print output at defined time-step intervals, rather than at a defined simulated time interval. For example, output may be written every 10 time steps, rather than every hour of simulated time. Time step lengths are calculated internally and depend on the flow conditions. For example, the simulation may require using very small time steps when a stress is imposed in the simulation, and these time steps may become larger, or “relax” over time. Printing intervals every set number of time steps therefore does not generally produce output at the same times for different runs, and coordination with observations is generally difficult and sometimes impossible. Output files may be of different lengths for different sets of parameter values; results from a given simulated time may not always appear in the same line in the output file. To create output files that are consistent in length, modify the output settings in HYDRUS as follows.

HYDRUS 1D Output Time Stepping

Output settings are found in the “Print Information” menu:



Print Information

Print Options

□ T-Level Information

Every n time steps: 1

☑ Print at Regular Time Interval

Time Interval: 0.16666

☑ Screen Output

☑ Hit Enter at End?

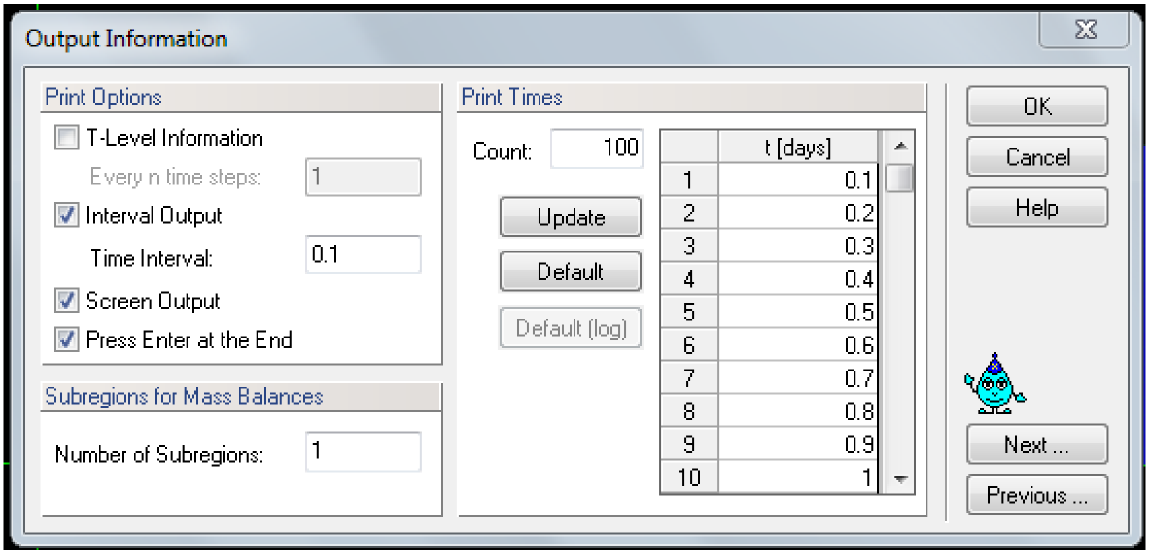
Print Times

Number of Print Times: 216

In this menu, un-check “T-Level Information” and instead check “Print at Regular Time Interval”. In the box next to “Time Interval” type the time interval for output, for example every 0.1 days. Notice that data printed according to “Print at Regular Time Interval” is different from the data printed according to “Print Times”. Generally, the data printed at simulation times set in “Print Times” is more detailed, whereas the data printed at simulation times set in “Print at Regular Time Interval” is mostly parameter averages. The user should refer to the HYDRUS 1D manual for a complete list of the types of data that are stored according to each of these options. Also, when observations are at different times that are not regularly spaced, it may be necessary to interpolate values to the observation time. This can be accomplished as described in the UCODE instructions for the Derived_Observations input block.

HYDRUS 2D/3D Output Time Stepping

The output settings are found in the menu “Output Information”:



Output Information

Print Options

☐ T-Level Information

Every n time steps: 1

☑ Interval Output

Time Interval: 0.1

☑ Screen Output

☑ Press Enter at the End

Subregions for Mass Balances

Number of Subregions: 1

Print Times

Count: 100

	t [days]
1	0.1
2	0.2
3	0.3
4	0.4
5	0.5
6	0.6
7	0.7
8	0.8
9	0.9
10	1

In this menu, un-check “T-Level Information” and instead check “Interval Output”. In the box next to “Time Interval” type the time interval for output, for example every 0.1 days. Notice that “Interval Output” is different from the “Print Times” table on the right-hand side of the menu. Generally, the data printed at simulation times set in “Print Times” is more detailed, whereas the data printed at simulation times set in “Interval Output” is mostly parameter values averaged across the entire domain. The user should refer to the HYDRUS 2D/3D manual for a complete list of the types of data that are stored according to each of these options.

Chapter 9: CALIBRATION AND ANALYSIS OF A GROUNDWATER MODEL FOR THE FAUX VALLEY SYNTHETIC FLOW SYSTEM: AN INSTRUCTIONAL REPORT

A common goal of modeling is to provide accurate predictions and clear guidance for future planning. What models do is apply conservation principles to conceptual models. Thus, models can quantitatively test ideas about properties and processes in a manner not available for complex systems by any other method. While this contribution to the toolbox of understanding natural systems is important, work over the last 30 years has shown that models typically do not achieve the level of prediction accuracy and skill that was idealized at one time. This leads to the question: What are reasonable goals for modeling environmental systems? We suggest that a major goal is to provide insights into system performance using data and models. To this end, the goal of model analysis is to increase model transparency and. Here, transparency means understanding what system features are likely to be important and unimportant, including identifying dominant parameters, important observations, and controlling processes. Testability means designing models and data collection such that as many aspects of the model as possible can be tested as meaningfully as possible, and is sometimes called falsifiability. A second way of looking at transparency is to identify what model aspects are and are not testable using available data.

In this Chapter, we use a model of a synthetic groundwater flow system to demonstrate how a report might be written that uses inverse methods and associated analyses to address the goals of transparency and testability. The advantage of using a synthetic system is that results simulated in a variety of ways can be compared to “true” values. While in this chapter we mostly seek to provide an example report, additional guidance and comments are also included. Generally, the content in a given report would be tailored more closely to the study conducted, and the additional comments would not be part of a published modeling report. Other recent examples of modeling reports include those distributed through the USGS Ground Water Resources Program (GWRP) and can be found at http://water.usgs.gov/ogw/gwrp/activities/gw-avail.html.

Introduction

This report describes an investigation of groundwater flow in the Faux Valley conducted using the U.S. Geological Survey groundwater flow model MODFLOW (Harbaugh, 2005) to simulate the system. The available observations include 10 heads and one measure of groundwater flow to the river under pre-stressed conditions.

For prediction, the following questions are considered for a future time after the flow system has adjusted to a proposed new pumping well:

(1) Will the drawdown at the southeast corner of the domain decline by more than 20 meters (m)? With 20 m of drawdown, local domestic wells will start to go dry and the local municipality will need to consider the expensive proposition of extending city water supplies to the area.

(2) Will the head at the pumping well decline to less than 80m? At this level the saturated thickness of aquifer 1 would be only 30 m at the well screen; less is considered to be problematic.
(3) Will the 2.2 m^3/s pumping rate cause groundwater flow to the stream to be less than 4.0 m^3/s, which is the minimum required flow?

Conceptual Model of the Groundwater Flow System and Hydrogeologic Framework

Groundwater flow in the system is thought to occur as illustrated in Figure 18a. Without pumpage, the main inflow is areal recharge and the main outflow is discharge to the stream.

The valley consists of a fluvial deposit underlain by a fine-grained lacustrine deposit, which is underlain a sequence of deltaic deposits derived from the bordering highlands. The bottom deltaic deposit is 50 m thick, the lacustrine deposit is 10 m thick, and the top layer is bounded above by a phreatic water table. There is geologic evidence of more transmissive material through the center of the surficial fluvial deposits and that the deeper aquifer increases in hydraulic conductivity with distance from the stream and proximity to the adjoining highlands.

Simulation of the Groundwater Flow System

The Faux Valley flow system is simulated using MODFLOW, which is composed of packages that can be used to simulate different parts of the system. The packages used to simulate this system are shown in Figure 18b. The model grid is shown in Figure 19 and is composed of 18 rows and 18 columns. All model cells are 1,000 m by 1,000 m. Thus, the simulated area is 18,000 m by 18,000 m.

The model layers are defined using the hydrogeologic layers shown in Figure 18. The results shown here are from final runs for which the top of the system is simulated as a phreatic water table. For initial model calibration runs, the top model layer thickness was approximated using the observed heads. Setting the thickness of free-surface model layers using observed heads is a useful way to reduce execution time during model calibration (Sheets et al., 2014).

This work considers two models of the Faux Valley flow system, which differ in how the hydraulic conductivity of the top model layer is parameterized. The models are steady state – reflecting calibration conditions with no pumpage and prediction conditions after pumping has occurred for an extended period of time.

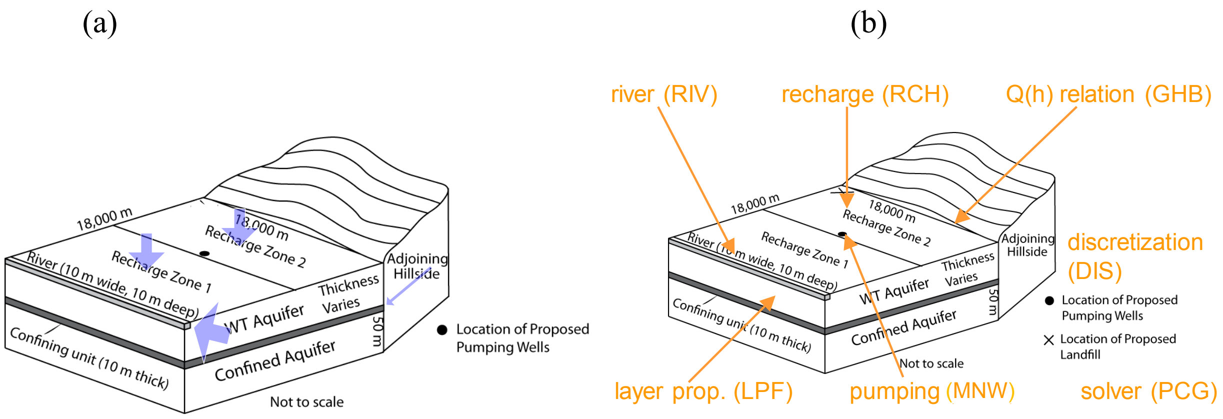


Figure 18. Conceptual model of (a) the groundwater system and (b) the MODFLOW Packages used to simulate its features, including River (RIV), Recharge (RCH), General-Head Boundary (GHB), Layer Property Flow (LPF), and pumping (MNW).

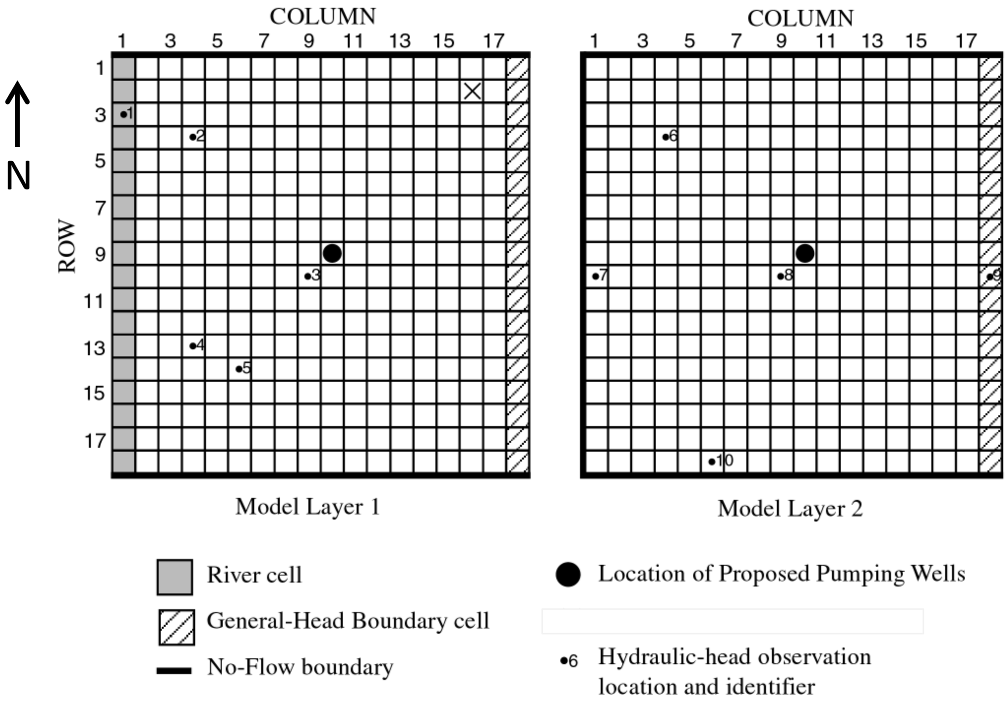


Figure 19. Model grid, location of features simulated using the MODFLOW River (RIV), General-Head Boundary (GHB), and Multi-Node Well (MNW) Packages, and location of the 10 wells with observed heads.

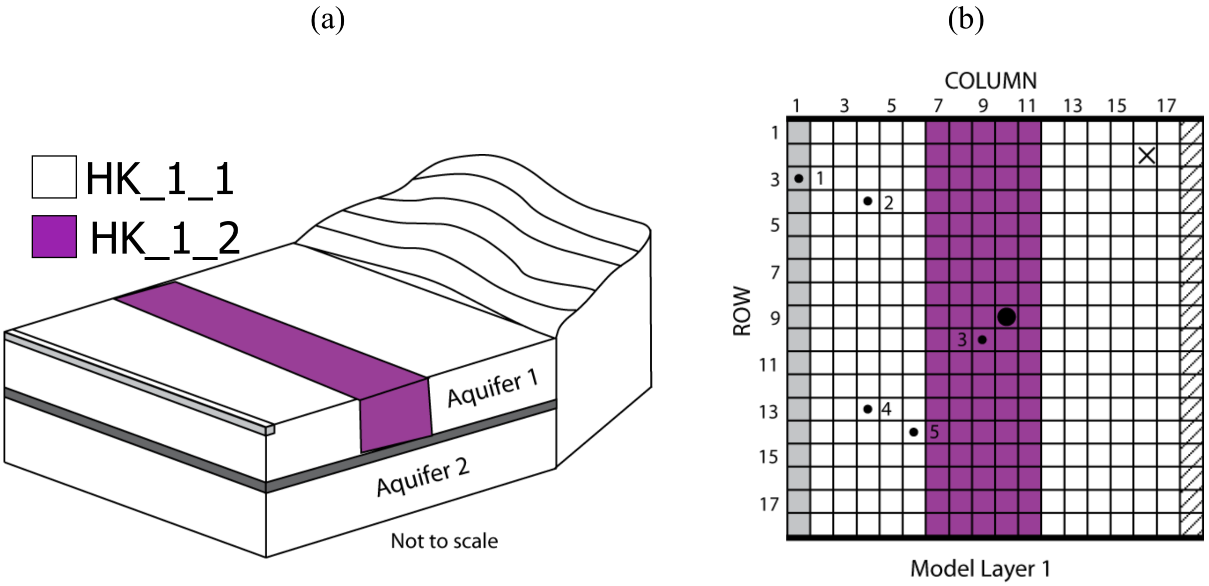


Figure 20. Definition of the hydraulic conductivity of model layer 1. (a and b) The two parameters defined for model B.

Model Calibration

The models are calibrated using UCODE; PEST was used to simulate a highly parameterized version of a similar system, as reported by Hill (2008). For the two alternative models considered, parameter values are modified to achieve the best match between observations and their simulated equivalents; independent information on the parameter values can also be included. The match is quantified using the following objective function:

$$
\begin{aligned}
S(\boldsymbol{b}) = \sum_{ih=1}^{nh} \omega_{ih}\left(h_{ih} - h'_{ih}(\boldsymbol{b})\right)^2 \\
+ \sum_{iq=1}^{nq} \omega_{iq}\left(q_{iq} - q'_{iq}(\boldsymbol{b})\right)^2 + \sum_{ip=1}^{npr} \omega_{ip}\left(P_{ip} - P'_{ip}(\boldsymbol{b})\right)^2 \\
+\beta \sum_{ir=1}^{nreg} \omega_{ir}\left(R_{ir} - R'_{ir}(\boldsymbol{b})\right)^2
\end{aligned}
\tag{10}
$$

The first term on the right includes head observations, the second includes flow observations, the third includes independent information on the parameters (referred to as prior information), and the fourth includes a different kind of independent information on the parameters, which is generally referred to as regularization. Sometimes prior information and regularization are combined in a single term; for highly parameterized models it often is useful to use different terms as shown here. The variables in equation 10 are defined as follows:

$\boldsymbol{b}$ is a vector of parameter values.
$S(\boldsymbol{b})$ is a symbol used to represent the objective function. It is written to show that the objective function value for a given model and a given set of observations and prior information changes as the parameter values change.

ih is an index that identifies a head observation.
iq is an index that identifies a flow observation.
ip is an index that identifies a piece of prior information.
ir is an index that identifies one regularization expression.

nh is the number of head observations.
nq is the number of flow observations.
npr is the number of prior information.
$nreg$ is the number of regularization expressions.

ω_{ih} is the weight for head observation *ih*.
ω_{iq} is the weight for flow observation *iq*.
ω_{ip} is the weight for prior information *ip*.
ω_{ir} is the weight for regularization expression *ir*.
β is a multiplier of all the weights on regularization.

h_{ih} is head observation *ih*.
q_{iq} is flow observation *iq*.
P_{ip} is prior information *ip*.
R_{ir} is the value of regularization *ir*.

$h'_{ih}(\boldsymbol{b})$ is the simulated equivalent of head observation *ih*.
$q'_{iq}(\boldsymbol{b})$ is the simulated equivalent of flow observation *iq*.
$P'_{ip}(\boldsymbol{b})$ is the simulated equivalent of prior information *ip*.
$R'_{ir}(\boldsymbol{b})$ is the simulated equivalent of prior information *ir*.

Equation 10 shows weighting that can be expressed as one multiplier for each additive quantity. In some cases, more complex weighting requires that a weight matrix be used. In such circumstances, equation 10 needs to be expressed in matrix notation as

$$S(\boldsymbol{b})=(\boldsymbol{y}-\boldsymbol{y}'(\boldsymbol{b}))^{\mathrm{T}}\boldsymbol{\omega}(\boldsymbol{y}-\boldsymbol{y}'(\boldsymbol{b})) + \beta(\boldsymbol{R}-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{\omega}_{\mathrm{r}}(\boldsymbol{R}-\boldsymbol{R}') \qquad (11)$$

where
$\boldsymbol{y}$ is a vector of observations and prior information.
$\boldsymbol{y}'(\boldsymbol{b})$ is a vector of simulated equivalents of the quantities in **y**.
$\boldsymbol{\omega}$ is a weight matrix on observations and prior information.
β is defined for equation 1.
$\boldsymbol{R}$ is a vector of regularization values.
$\boldsymbol{R}'$ is a vector of regularization equations which are a function of the parameter values.
$\boldsymbol{\omega}_{\mathrm{r}}$ is a weight matrix on regularization values.

Equation 11 is equivalent to equation 10 if the weight matrices $\boldsymbol{\omega}$ and $\boldsymbol{\omega}_r$ are diagonal – that is, only terms along the diagonal are nonzero. For the problem discussed in this section equation 10 is used.

Adequacy of the calibrated model is tested based on (1) model fit to observations, (2) reasonableness of estimated parameters, (3) whether observations and parameters important in the calibration seem appropriate given the conceptual model and model construction, (4) whether the parameters important to the predictions of interest are informed by the available observations and prior information. The importance of observations and parameters is evaluated using sensitivity analysis.

If highly parameterized methods such as pilot points are used to parameterize a model property such as hydraulic conductivity, the overall model fit is often prescribed, and thus model fit becomes a less useful indicator of model adequacy. It thus becomes very important to carefully consider the other measures of model adequacy when using highly parameterized methods.

The sensitivity results presented in this report evaluate the importance of observations to defined parameters, the importance of parameters to the fitting of observations, the importance of parameters to predictions, and the importance of observations to predictions. Results are presented for two types of sensitivity analysis: local methods and a "spider plot" produced by a basic sensitivity analysis conducted by increasing and decreasing the parameter values by a defined amount. The spider plot is presented because some readers may find it easier to understand.

Observations and Weighting

The calibration observations are based on measurements taken before the onset of pumping. They reflect long-term average unstressed conditions, and are considered to be at steady-state. The observations include one head measurement from each of the ten wells (Figure 19 and 20b) and one flow. The head observations are distributed spatially, and are referred to as hd01.ss through hd10.ss. The flow observation is the measured gain in flow from the groundwater system along the entire length of the river and is referred to as Flow01_1.

The weighting of observations and prior information in equations 10 and 11 serves two complementary purposes. First, the weighting makes each term dimensionless, so that summing them into one scalar, $S(\boldsymbol{b})$, is meaningful. Second, a larger weight makes a given term more important in the objective function. As discussed by Hill and Tiedeman (2007, guideline 6 and Appendix C), these purposes and the theoretical requirement for the lowest variance parameter estimates are all satisfied if the weighting is error-based, meaning that it is based on an analysis of errors in the observations. Like many other works (for example, Mroczkowski et al., 1997; Oliver et al., 2008; Foglia et al., 2009, Renard et al., 2011; Tiedeman and Green, 2013), this work uses error-based weighting.

In error-based weighting, the weights of equation 10 equal the reciprocal of the error variance. In UCODE, the user can input the weight, square-root of the weight, variance, standard deviation, or coefficient of variation. The program will use the provided information to calculate the weight. To convert the coefficient of variation to a weight requires that it be multiplied by the related observed or simulated value. There is some debate about whether the best value to use is observed or simulated; in the runs for this chapter observed values are used. Chapter 3 of this documentation describes the enhanced capabilities UCODE_2014 provides when a coefficient of variation is used.

For the Faux Valley study, the sources of error in heads include errors in measuring the water level in the well (standard deviation about 0.015 m) and errors in the elevation of the well head (standard deviation 0.33 m). To add these sources of errors, square the standard deviations to obtain variances and add the variances (it is not accurate to add standard deviations). This produces head observation variances of 0.109225 m^2, standard deviations of 0.33, and weights of about 10.

The flow gain of 5.77 m^3/s is measured by calculating the difference in upstream (4.80 m^3/s) and downstream (10.57 m^3/s) measurements of streamflow. Error in each of these streamflow measurements contributes to the error in the flow gain. Carter and Anderson (1963) suggest that streamflows can be measured with errors that have standard deviations equal to about 5% of the flow under ideal circumstances. When errors are expressed as a percent, the associated fraction, here 0.05, is the coefficient of variation. Using Carter and Anderson (1963) as guidance, a coefficient of variation for the flow gain is calculated from the coefficients of variation for the streamflows. The standard deviations of the upstream and downstream measurements are 0.05×4.80 m^3/s=0.24 m^3/s and 0.05×10.57 m^3/s=0.53 m^3/s, respectively. Squaring and summing these standard deviations yields a total variance of 0.34 m^6/s^2 for the flow gain. Taking the square root and dividing by the 5.77 m^3/s flow gain yields a coefficient of variation of 0.10.

For the UCODE_2014 input files, the head standard deviation of 0.33 m and the flow coefficient of variation of 0.10 are entered and the weight is calculated by the program.

If PEST were being used, the user must enter the weights in the input file. From an error-based perspective, PEST defines a weight as the reciprocal of the error standard deviation, so that it equals the square-root of the weight in UCODE. Thus, in PEST the value to enter for the weight for each head observation would be 1/0.33049=3.026. The value to enter for the weight of the flow observations would be 1/(5.77×0.10)=1.73, where 5.77 is the observed flow.

Parameters

Two alternative conceptual models are considered for the Faux Valley, and are denoted models A and B. They differ in how the hydraulic conductivity of the water table aquifer is parameterized. Model A has seven defined parameters; model B has eight defined parameters. All of the defined parameters are described below, starting with those that are the same in all models. The parameters are described in the context of the MODFLOW Package in which they are defined.

Parameters Common to all Models

For all models, three parameters are defined in the Layer Property Flow (LPF) Package to produce the hydraulic-conductivity distribution of the confining unit and confined aquifer. Parameter HK_2 multiplies a geologically based linear variation in model layer 2 that results in the simulated hydraulic conductivity increasing from the value defined under the river, which equals HK_2, to nine times HK_2 at the system boundary farthest from the river. Parameter VK_CB equals the hydraulic conductivity of the confining bed. VANI equals the vertical anisotropy of layers 1 and 2.

Two parameters are defined in the Recharge (RCH) Package. Each defines the areal recharge rate in one of the two recharge zones shown in Figure 18. Parameter RCH_1 equals the recharge in zone 1. Parameter RCH_2 equals the recharge in zone 2.

One parameter is defined in the River (RIV) Package, and is named K_RB. To explain K_RB, consider that the River Package calculates flows between the groundwater system and the river as

$$q_i=C_i(HRIV_i - h_i) \qquad (12)$$

where

q_i is the calculated flow for grid cell i.

C_i is the conductance that relates a head difference to a flow rate.

$HRIV_i$ is the specified water level in the river in grid cell i.

h_i is the head in the groundwater system adjacent to the river in grid cell i.

Parameter K_RB is used to define the conductance as

$$C_i=(K\_RB \times area_i)/thickness_i \qquad (13)$$

where $area_i$ is the area of the river within grid cell i. While this formulation suggests that the value of K_RB can be related to the hydraulic conductivity of the river bed, Mehl and Hill (2010) suggest that the value is grid-scale dependent. This means that if the grid is refined or made coarser at the river, K_RB would need to be recalibrated.

Parameters that Vary between Models

Model A has one parameter used to control a homogeneous hydraulic-conductivity distribution in the top model layer. The hydraulic conductivity of this layer equals the value of parameter HK_1 in the MODFLOW LPF Package and the UCODE main input file. Calibration efforts began with model A and evidence of significant model error was revealed by misfit to observations and unrealistic estimated parameter values. This led to development of model B. Some sensitivity analysis results from model A are provided in this report.

Model B differs from model A in that two hydraulic-conductivity parameters are used to define the distribution in model layer 1, as shown in Figure 20. Parameter HK_1_2 covers the region where the river was likely situated earlier in the Holocene Epoch. The sediments in this region are thought to be coarser than those in the region to which parameter HK_1_1 applies.

Prior information

Independent information exists for all of the parameters, as shown in Table 5.

Table 5. Starting values and lower and upper reasonable values of model B parameters, based on field information.[1]

Parameter	Lower reasonable value[2]	Starting value (also used as prior value)	Upper reasonable value[2]	Log transform (base 10 of prior value, if applicable)	Standard deviation designated for prior (applies to log transform if applicable)
HK_1 (m/s)					
Low K area	9.6×10^{-5}	3.0×10^{-4}	9.5×10^{-4}	-3.52	0.25
High K area	9.6×10^{-4}	3.0×10^{-3}	9.5×10^{-3}	-2.52	0.25
HK_2 (m/s)	4.0×10^{-6}	4.0×10^{-5}	4.0×10^{-4}	-4.40	0.5
VANI (---)	0.8	1.0	1.2	--	0.1
VK_CB (m/s)	1.0×10^{-8}	1.0×10^{-7}	1.0×10^{-6}	-7.00	0.5
K_RB (m/s)	1.2×10^{-4}	1.2×10^{-3}	1.2×10^{-2}	-2.92	0.5
RCH_1 (cm/yr)	16	32	48	--	8
RCH_2 (cm/yr)	47	63	79	--	8

[1] In a real report field information needs to be described to justify the listed values.

[2] Here, lower and upper reasonable values are calculated by, respectively, subtracting and adding two standard deviations to the starting parameter value. For log transformed parameters, calculations are performed using transformed values and then converted to native values.

The information in Table 5 can be used in three ways for model calibration (Hill and Tiedeman, 2007, p. 288-290). The approach used generally depends on whether or not the observations provide ample information on the parameter, as indicated using the sensitivity analysis graphs presented in the following section of this report. Inadequate information can result from parameter insensitivity or correlation. For prediction uncertainty evaluation, the best way to include the prior information may differ from that used in model calibration. The three options for using the information are:

1. Calibrate without defined prior information. Compare the upper and lower reasonable values with estimated parameter values to check whether the estimated values are realistic. Unrealistic parameter values can indicate model error.
2. Include the independent information as prior information in the objective function. This is a Bayesian approach to model calibration.
    a. In regression, the advantage of depending on prior information instead of parameter constraints is to produce parameter estimates that are close to a central value instead of at the extremes of the range of likely values.
    b. If the observations contain abundant information about the parameter, comparing options 1 and 2 provides a way to evaluate whether there is a contradiction between the prior information and the observation data. Larger differences in the parameters estimated by the two different options indicate greater contradiction and greater likelihood of model error or some problem with the prior information and how the parameters are defined.
    c. Option 2 often is used to enforce reasonable values of parameters for which unrealistic values are estimated when using option 1. The existence of these errant parameter values is considered to indicate model error. Adding the prior information,

and not addressing the possible model error, may or may not enhance the predictive skill of the model. This important concern is beyond the scope of the present work.

3. Use the prior information to set the parameter value and exclude the parameter from the regression.
    a. This can make the regression runs shorter and can be a useful approach if the parameter is provided inadequate information from the observations, as indicated by small values of composite scaled sensitivity (CSS) and(or) large absolute values of parameter correlation (PCC).
    b. Option 3 is useful for model calibration but is not advisable for parameter or prediction uncertainty analysis. When quantifying uncertainty, parameters need to be reactivated and prior information added if available. This allows the uncertainty of the parameter to be appropriately accounted for when evaluating prediction uncertainty.

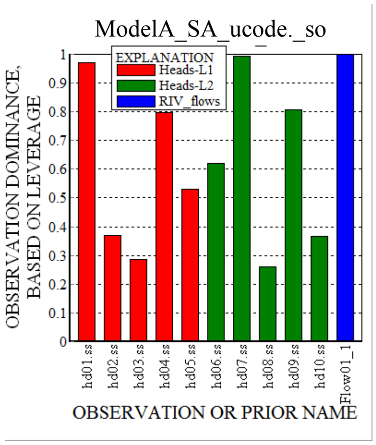
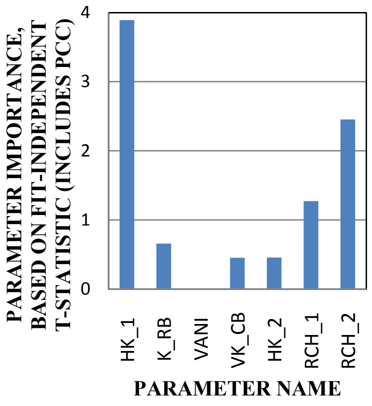
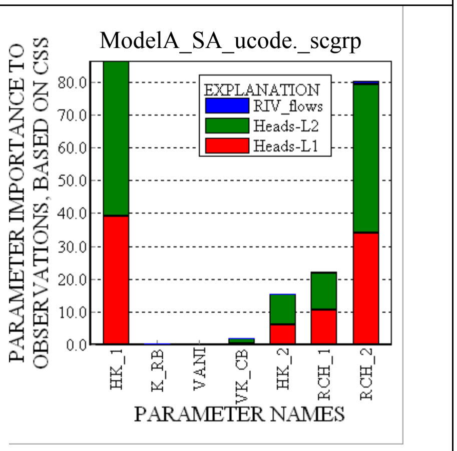
Initial Sensitivity Analysis to Identify Important Observations and Parameters

Regression and model fit are often dominated by a limited number of observations and a few of the defined parameters. Identifying important observations and parameters can help guide model calibration and provide insight about simulated processes. Also, it can be used to focus additional effort on reviewing or improving the observation measurements, the conceptual model, and the model parameter definition.

The initial sensitivity analysis for the Faux Valley model was conducted using model A (with homogeneous model layer 1) and the starting parameter values from Table 5. Dominant observations and important parameters were identified using leverage, fit-independent parameter t-statistics, and CSS (defined in the section “Definition of Statistics, Including Equations” at the end of this report), as shown in Table 6. The leverage results suggest the dominance of the streamflow gain observation and the heads in layers 1 and 2 under the river (hd01 and hd07), and, to a lesser extent, hd04 and hd09. The CSS and t-statistics show that the hydraulic conductivity of the top model layer and the two recharge rates are the most important parameters. These measures also suggest that the available observations do not support estimating the value of VANI, and might not support estimation of K_RB, VK_CB, and HK_2.

On the basis of the leverage results, additional field work was conducted to confirm that boreholes 01, 04, 07, and 09 (Figure 19) were constructed and operating as expected. For example, borehole optical televiewer logs were run to confirm that the well screens were open to the reported depths and were not clogged.

Table 6. For model A, important observations identified using leverage and important parameters identified using the fit-independent version of the parameter t-statistic and CSS. In all graphs, larger bars identify more important quantities.

Question, Answer, and Action	Graph[1,2]
Question: Which observations potentially dominate one or more of the estimated parameter values? **Answer:** Flow01_1 and hd07.ss, with leverage equal to 1.0, are potentially important to one or more of the estimated parameter values. Observations hd01.ss, hd04.ss and hd09.ss also have high leverage. **Action:** Check for and reduce errors in the measurements and simulation related to these important observations.	
Question: Which parameters are most important to fitting the observations? Equivalently, which parameters are provided the most information by the observations? **Answer:** HK_1, RCH_2, and RCH_1 are the most important parameters. **Action:** If additional parameters will be defined, consider aspects of the system related to the most important parameters. For example, HK_1 might be replaced with more than one parameter.	
Question: Which observation groups contribute knowledge to each parameter? **Answer:** The heads in the lower and upper aquifers contribute to each of the important parameters similarly. The small contribution of the flow observation contradicts its large importance as indicated by leverage. This suggests the importance of the flow is in reducing parameter dependence (correlation), because leverage accounts for this dependence whereas CSS does not. **Action:** The flow contributes to the unique estimation of many parameters. Checking and improving the flow observation and its simulation is a high priority.	

[1] Equations for the plotted quantities are provided in the section "Definition of Statistics, Including Equations".
[2] Leverage and t-statistic include the effects of parameter correlation (parameter interactions) as determined by linear theory. CSS does not.

Use of Regression to Estimate Parameter Values

UCODE was used to estimate parameters by a modified Gauss-Newton (Levenburg-Marquardt) method. For Model A, the value of VANI was fixed at its starting value because it is not informed by the observations (Table 6). One recurring problem during the regression was that the estimate of recharge near the hillside, RCH_2, was consistently smaller than the estimate of recharge near the river, RCH_1. This is in contrast to infiltration data showing that in the Faux Valley recharge is larger near the hillside (see range of reasonable values in Table 5).

When these difficulties were encountered with calibration of model A, alternative conceptual models were considered. The importance of HK_1 in initial model A (Table 6) and similar results for calibrated model A suggested that the observations provided enough information to represent layer 1 using more than one parameter. The geologic information discussed in the “Parameters” section led to development of model B, which represents layer 1 with two hydraulic conductivity zones (Figure 20).

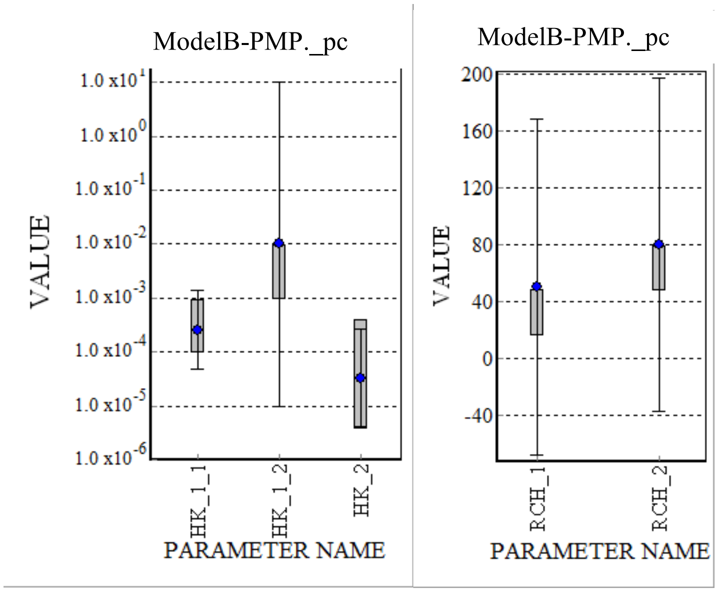
For model B, the values of unimportant parameters VANI, K_RB, and VK_CB were also fixed during the calibration. An alternative is to add prior information for these parameters, but setting the values reduced the calibration execution time. This was of little consequence for this small problem and is demonstrated here because it can be greatly advantageous for larger models. An attempt at estimating the remaining five parameters (HK_1_1, HK_1_2, HK_2, RCH_1, RCH_2) produced poor regression performance. The CSS calculated at the initial values for these parameters showed that HK_1_2 was least important (Appendix D, Table D-3, column 1). Therefore, prior information was added on this parameter, using its starting parameter value shown in Table 5, which led to a tractable regression. The estimated parameter values for model B are listed in Table 7.

Analysis of Calibrated Model Fit and Parameter Values

The calibrated model is analyzed by evaluating the parameter estimates, the model fit to observations and measures of leverage, influence, and parameter importance.

The parameter estimates are compared to reasonable parameter ranges in Table 7. The estimates for HK_1_2 (High K area of HK_1), RCH_1, and RCH_2 are each slightly larger than the upper limit of their respective reasonable range. This suggests a mild degree of model error. Notably, HK_1_2 has prior information (recall that it was needed to obtain convergence of the regression) and the prior value is at the center of the reasonable range, but the HK_1_2 estimate is not close to that value. The poor fit to the prior is related to the circumstance that both the prior and observation Flow_01_1 reduce the extreme correlation of all model parameters except VANI. This is discussed below in the context of Figure 21.

Table 7. Estimated parameter values (blue circles on figures), reasonable ranges (wide gray bars; from Table 5), and 95% linear individual confidence intervals (black lines) for model B.

Parameter	Estimated or fixed value	
HK_1 (m/s)		
Low K area	2.60×10^{-4}	
High K area	1.13×10^{-2}	
HK_2 (m/s)	3.18×10^{-4}	
VANI[1] (---)	1.0	
VK_CB[1] (m/s)	1.0×10^{-7}	
K_RB[1] (m/s)	1.2×10^{-3}	
RCH_1 (cm/yr)	49.8	
RCH_2 (cm/yr)	80.7	

[1] Parameter is fixed in regression.

Parameter uncertainty is modest and consistent with prior information for HK_1_1 and HK_2 (Table 7). In contrast, uncertainty is very large for HK_1_2, RCH_1, and RCH_2, indicating these estimates are not well constrained by the available observations. For the recharge parameters, the lower limits of the confidence intervals are negative. Such unrealistic limits can result from using linear confidence intervals, which are defined simply as the estimated parameter value plus and minus a term that is a function of the parameter standard deviation (Hill and Tiedeman, 2007, p. 138). Nonlinear confidence intervals (Hill and Tiedeman, 2007, p. 139) or Markov chain Monte Carlo (MCMC) methods (Chapter 5) are alternatives for computing uncertainty that use model simulations to explore parameter space, and generally do not result in unrealistic confidence limits. Nonlinear confidence intervals require about ten times more model runs than the linear confidence intervals, while MCMC requires orders of magnitude more runs.

Model fit is assessed by evaluating components of the objective function defined in equation 10. This includes comparing observations and their simulated equivalents, and prior information values and their estimated equivalents. It also includes evaluating the regularization, if used, including the value of β.

For model B, Table 8 shows that the standard error (s) equals 7.6 and its confidence interval does not include 1.0. An advantage of using error-based weighting is that this provides a meaningful test of model fit, as follows. If the model fit is consistent with the assumptions used in specifying the weights on the observations, and the model is a correct representation of the true Faux Valley flow system, then the expected value of s is 1.0. Because s is larger than 1.0 and its entire confidence interval also is greater than 1.0, the model fit is significantly worse than expected given the assumptions associated with the weights. Given that the statistics used to calculate the weights are thought to accurately quantify the error in the observed heads, flows, and prior information, the larger-than-expected value of *s* indicates significant model error. The analysis of the weighted residuals can help with determining the sources of this error.

Table 8. Fit to observations for model B.

Observation	Observed	**Model B (PMP)**		
Name	value	Simulated	Residual	Weighted Res.
hd01.ss	100.25	100.31	-0.06	-0.19
hd02.ss	149.41	148.70	0.71	2.16
hd03.ss	163.46	164.43	-0.97	-2.93
hd04.ss	153.75	148.70	5.06	15.32
hd05.ss	160.36	163.76	-3.40	-10.31
hd06.ss	148.90	148.53	0.37	1.13
hd07.ss	103.77	103.28	0.49	1.47
hd08.ss	163.87	164.55	-0.68	-2.07
hd09.ss	180.67	179.33	1.34	4.06
hs10.ss	160.80	162.27	-1.47	-4.46
Flow01_1	-5.77	-6.75	0.98	1.69
Measures of model fit				
Standard error[1] and its (confidence interval)		7.6 (5.0;15.5)		
[2]$S(\boldsymbol{b})$	[5]21,896	406		
[3]$S(\boldsymbol{b})_{\mathrm{obs}}$	[5]21,896	401		
[4]$(S(\boldsymbol{b})_{\mathrm{obs}})/(nh+nq))^{1/2}$	32	6.1		

[1] The unbiased estimate of the standard error, calculated as $[S(\boldsymbol{b})/(n\text{-}p)]^{1/2}$. *n* is the total number of observations and prior information (*nh*+*nq*+*npr* for eq. 10; here the value is 12). *p* is the number of parameters estimated, which is 5. Confidence intervals are calculated as described by Hill and Tiedeman (2007, p. 96, eq. 6.2).
[2] Calculated as shown in eq. 10.
[3] Calculated with observations only and no prior.
[4] Biased estimate of the standard error that is commonly presented; provided here for comparison. It is always smaller than the unbiased estimate of the standard error.
[5] Calculated for the starting parameter values in Table 5. $S(\boldsymbol{b})$ equals $S(\boldsymbol{b})_{\mathrm{obs}}$ because the starting value of HK_1_2 equals its prior value.

The fit to individual observations and prior information is examined using weighted residuals (Table 8 and Figure 21). Ideally, weighted residuals are randomly distributed when plotted against simulated values (Figure 21b) and on maps of the model domain (Figure 21c). Non-randomness, or bias, is an indication of model error. For the Faux Valley model, there are very few observations. This makes assessment of randomness difficult, because a small sample size from a truly random distribution can appear nonrandom. The weighted residuals for the heads and prior appear to be evenly distributed above and below a value of 0.0 on Figure 21b, although there may be some bias indicated by head weighted residuals that are mostly negative (simulated values larger than observed values) for simulated values between about 160 m and 170 m. This is also apparent on Figure 21c, where in both layers, weighted residuals are mostly positive near the river, negative in the central columns of the model, and positive close to the hillside. In both layers, most of the negative residuals are in an area corresponding to that where HK_1_2 applies in layer 1. This suggests that there might be error in model B related to how zones are used to represent hydraulic conductivity in layer 1, and raises the question of whether zones are the best parameterization method. Further field characterization of the fluvial aquifer could be conducted to guide improved representation of this hydraulic property.

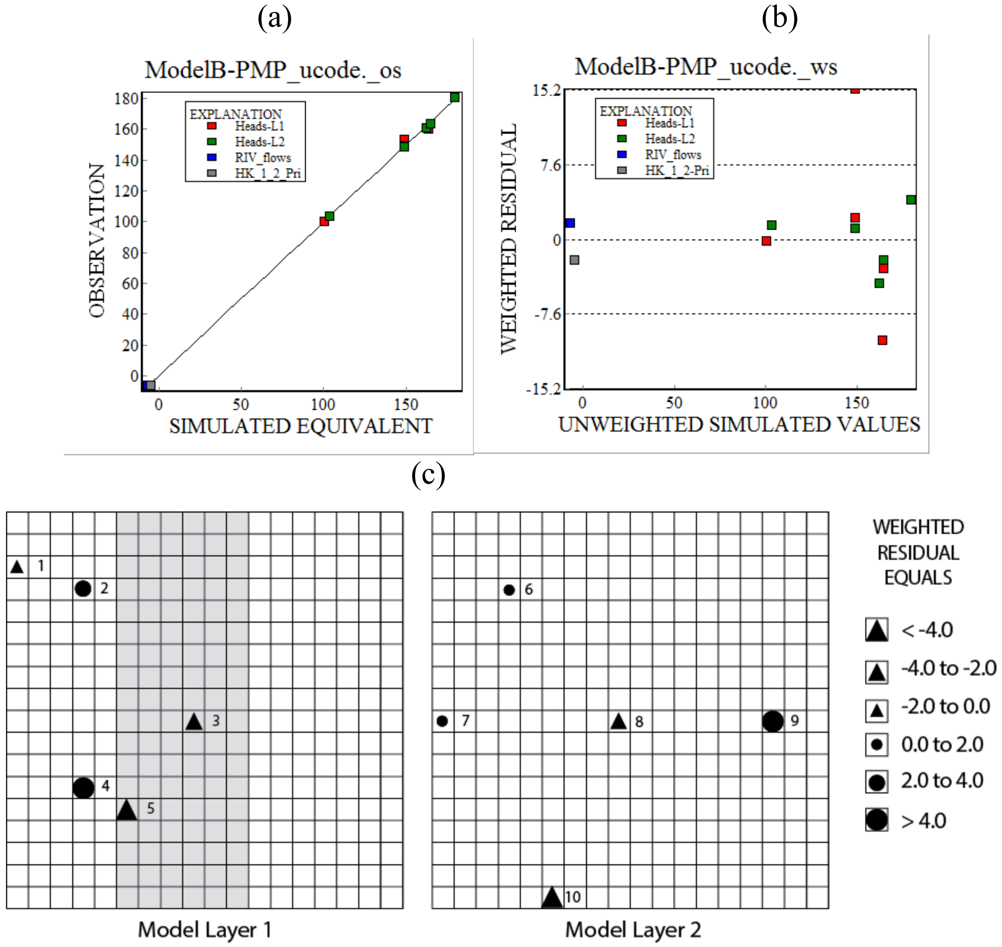


Figure 21. Model B: (a) Observed versus simulated values, (b) weighted residuals and unweighted simulated values, and (c) spatial distribution of weighted residuals over the model domain. In model layer 1, the unshaded model cells are where HK_1_1 applies and the gray shaded cells are where HK_1_2 applies.

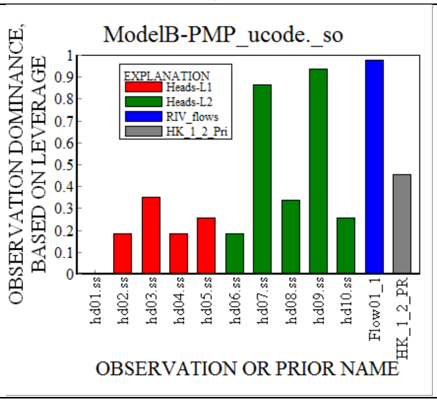
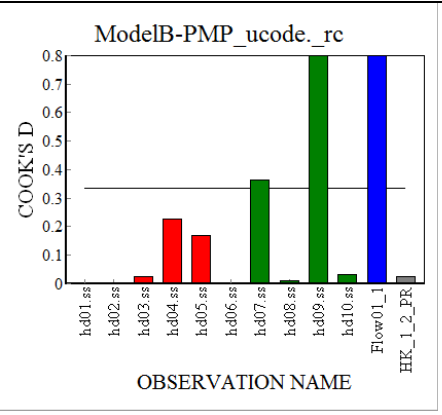
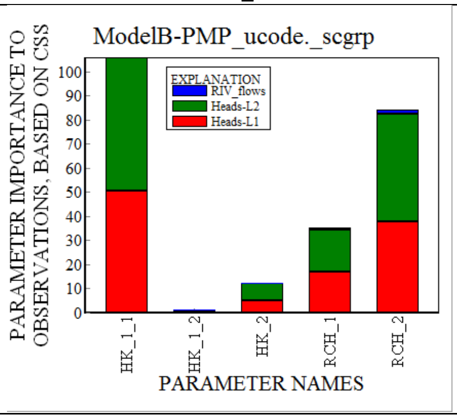
In the calibration, the flow observation and the prior information on HK_1_2 both reduce the extremely high parameter correlations that would occur if head observations alone were used to calibrate the model. The fit to the flow and the prior reflect this role. The regression minimizes their contributions to the objective function by estimating parameter values that produce a negative weighted residual for one of these data (the prior), and a positive weighted residual of similar magnitude for the other (the flow) (Figure 21b).

Leverage and Cook's D (Hill and Tiedeman, 2007, p. 134-136) can help identify individual observations or prior information that dominate the parameter estimates (Table 9). Flow01_1, hd09.ss, and hd07.ss are each potentially (identified by leverage) as well as actually (identified by Cook's D) important to the parameter estimates. Flow01_1 is influential because of its role in preventing extreme parameter correlations, and hd09.ss is influential because of its proximity to the hillside, where simulated heads are generally more sensitive to the hydraulic conductivity and recharge parameters than at locations near the river.

HK_1_2_PR has moderate leverage because this prior information potentially controls the value of parameter HK_1_2. However, the actual influence of HK_1_2_PR is much less than its potential influence. This prior information is important to achieving a stable regression, but the particular prior value specified does not appear to strongly affect the parameter estimates. This is consistent with the result that the estimated value of HK_1_2 (1.13×10^{-2} m/s) is not close to the prior value (3.0×10^{-3} m/s).

Comparison of the leverage for model A initial parameter values (Table 6) and for model B optimal estimates (Table 8) illustrates that by fixing the values of three parameters in the regression for model B, the importance of hd01.ss is diminished. Head observation hd01.ss is directly beneath the river (Figure 21c) and has high leverage when all parameters were active (Table 6) because it dominates estimation of K_RB. With the value of K_RB fixed in the model B regression, hd01.ss no longer plays that role.

Table 9. For calibrated model B, observations important to the estimated parameters identified using leverage and Cook's D, and estimated parameters most important to fitting the observations identified using CSS. Parameters fixed in the regression (VANI, VK_CB, and K_RB) are not included in this evaluation. In all graphs, larger values indicate greater importance.

Question, Answer, and Action	**Graph**[1,2]
Question: Which observations are potentially important to the estimated parameter values? **Answer:** Top three: Flow01_1, hd09.ss, hd07.ss. **Action:** Check for and reduce errors in the measurements and simulation related to these important observations.	
Question: Which observations are actually important to the estimated parameter values? (Actual in that the observed values are considered.) **Answer:** Flow01_1, hd09.ss, hd07.ss. **Action:** Same as for leverage, though hd07.ss is less important than the leverage suggested it might be.	 The horizontal line equals the critical value above which an observation is considered important. Cook's D for hd09.ss and Flow01_1 are 13.5 and 22.2.
Question: Which observation groups contribute information about each parameter? **Answer:** The heads in the lower and upper aquifers contribute similarly to each of the important parameters. The small contribution of the flow observation contradicts its large importance as indicated by leverage and Cook's D. This suggests the importance of the flow is in reducing parameter dependence (correlation).	

[1] Equations for the plotted quantities are provided in the section "Definition of Statistics, Including Equations".
[2] Leverage and Cook's D include the effects of parameter correlation (parameter interactions) as determined by linear theory. CSS does not.

Model Limitations

The model analyses suggest that model B is limited in its ability to predict quantities that depend on the vertical anisotropy, hydraulic conductivity of the riverbed, vertical conductivity of the confining bed, and the hydraulic conductivity HK_1_2 in the upper aquifer. These aspects of the system are not well supported by the available observations, and so the values used in the model for these parameters depend mostly or entirely on independently determined estimates. Quantities of interest that are sensitive to these parameters might not be simulated reliably with model B. Such quantities could include, for example, advective transport of chemical constituents in recharge water that migrate from the upper aquifer through the confining bed into the lower aquifer, and then migrate back to the upper aquifer and discharge through the riverbed.

The hydraulic conductivity HK_1_1 in the upper aquifer, the hydraulic conductivity of the lower aquifer, and recharge are provided more information by the observations. However, the final estimates of some parameters are at or a little beyond the limits of reasonable ranges. This suggests caution be exercised in using model results, and that collection of additional data to improve the model would be helpful. Strategies for such data collection are discussed in the section “Supplementary Model Analyses.”

The models are developed using head and flow data. Generally, models calibrated with one type of observation are not able to simulate other kinds of processes well. For example, this model may not be able to simulate transport very well, in part because there are no calibration observations that provide information about porosity parameters.

Prediction of Head, Drawdown, and Flow

As noted in the Introduction, the following water resources management questions are considered for a future time after the flow system has adjusted to pumping 2.2 m^3/s in a proposed new production well:

(1) Will the drawdown at the southeast corner of the domain decline by more than 20 m? With 20 m of drawdown, local domestic wells will start to go dry and the local municipality will need to consider the expensive proposition of extending city water supplies to the area.
(2) Will the head at the pumping well decline to less than 80 m? At this level the saturated thickness of aquifer 1 would be only 30 m at the well screen; a head less than the 80 m level is considered to be problematic.
(3) Will the 2.2 m^3/s pumping rate cause groundwater flow to the stream to be less than 4.0 m^3/s, which is the minimum required flow?

Model B is used to simulate the effect of the proposed pumping, and to analyze the predicted drawdown, head, and flow corresponding to the management questions. The pumping well is at row 9, column 10 of the model grid (Figures 18 and 19) and the desired pumping rate is 2.2 m^3/s. The MNW Package is used to represent pumpage, so the simulated pumping rate could be less than this value, but in the simulations discussed here the full rate is achieved. For convenience, in this synthetic problem the predicted head in the pumping well is assumed to equal the head simulated in the model cell containing the well, so that the OBS package of MODFLOW-2005 can be used to define this prediction. The MNW Package calculates the in-well head, and this simulated quantity could alternatively be used as the prediction.

The predictions for model B are listed in Table 10. All predictions meet the management criteria. These predictions are uncertain in part because there is uncertainty in the parameter values. This uncertainty can be propagated to uncertainty in the predictions, through the model. Prudent use of the model to help guide decisions about the proposed pumping well includes considering not only the predicted values, but also the uncertainty in these values. Next, the model is used to calculate the prediction uncertainty caused by parameter uncertainty.

Table 10. Predicted values for model B under pumping conditions All predictions meet the associated management criterion.

Description of prediction	Management criterion. Prediction needs to be…	Model B
Drawdown in the southeast corner, in meters	less than 20	17
Head at the well, in meters	greater than 80	144
Flow to the stream, in m^3/s	less than -4.0 (absolute value greater than 4.0)	-4.5

Prediction Uncertainty

For the prediction uncertainty analysis, all model B parameters are activated and prior information is specified on all parameters, so that realistic parameter uncertainty is translated to the calculation of prediction uncertainty. The prior values and associated standard deviations are listed in Table 5. The prediction uncertainty measures therefore reflect the information the observations provide about the parameters as well as all prior information. Linear confidence intervals on the predictions are shown in Figure 22. The results of Lu et al. (2012) suggest that linear intervals may provide a reasonable measure of uncertainty in many groundwater flow problems.

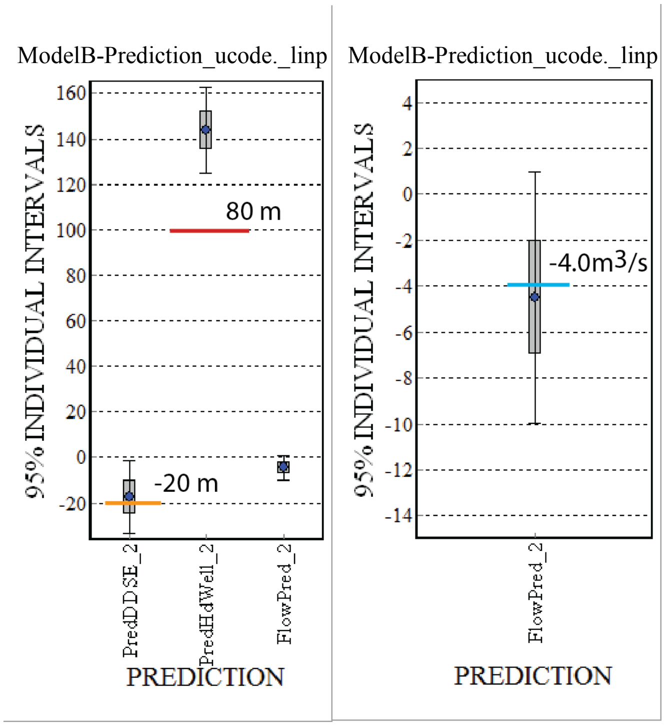


Figure 22. Predictions from model B (dark blue circles), their linear individual 95 percent confidence intervals (black lines) and plus and minus one standard deviation (wide gray bars), and the management criteria noted as colored lines. (a) Results for all predictions. (b) Results for FlowPred_2 expanded so that the confidence interval ranges are more visible. In MODFLOW, flow out of the groundwater system (here, flow to the river) is negative, so the management criterion is that flow must be less than -4.0 on the graph (greater than 4.0 m^3/s must flow to the stream).

The prediction uncertainties raise a serious concern with respect to decisions about drilling the new pumping well. For the drawdown and flow predictions, about half of their respective confidence intervals violate the management criteria, and for the flow prediction, the confidence interval is especially large. Given these results, prudent next steps might include a phased implementation of the pumping that includes monitoring of the streamflow and drawdown or collection of additional field data to improve model predictions. Collecting such data, and using it in the modeling analyses, could help increase confidence that implementing the new pumping would not cause too much drawdown or depletion of streamflow. Additional prediction analyses can be conducted to identify what new field data would most reduce the prediction uncertainty. For the flow system considered here, these analyses are described in the section "Supplementary Model Analyses."

Summary and Conclusions

A model of the Faux Valley groundwater flow system was developed to evaluate the effect of pumping in a proposed new production well that would tap both the upper and lower aquifers in the valley. The flow system with this pumping must comply with several management criteria, which stipulate that there be at least 4.0 m^3/s discharge from the upper aquifer to the river, and that the water table at the pumping well and in the southeast corner of the valley be greater than specified minimum levels.

Calibration and analysis of the Faux Valley model showed that the available hydraulic head and river discharge observations support estimation of recharge and of horizontal hydraulic conductivities in some portions of the aquifer, but do not support estimation of other system properties, including the vertical hydraulic conductivity of the confining bed and the riverbed.

The calibrated model was used to simulate the proposed pumping and to predict flow, head, and drawdown corresponding to the management criteria. Results indicated that all the predicted quantities met the associated management criteria. However, including the 95-percent confidence intervals for these predictions in the analysis revealed a high degree of uncertainty in the flow and drawdown predictions, and the possibility that these predictions would violate the criteria.

The prediction analysis prompted use of the model to identify potential new observation data and flow system characterization activities that would help to reduce the prediction uncertainty and lead to a more definitive conclusion about whether the management criteria would be violated. This evaluation is described in the "Supplemental Model Analyses" section, and showed that it would be beneficial to collect new head data at several existing wells, and a new river discharge observation, under conditions of pumping at a modest rate in existing monitoring well 03. These potential new observations ranked as important to all three predictions of interest. The model evaluation also showed that among the parameters, further characterization of the hydraulic conductivity HK_1_1 in the upper aquifer would most reduce the prediction uncertainty. The next steps for this study will be to collect the new observations and new aquifer characterization data identified as most important to the predictions.

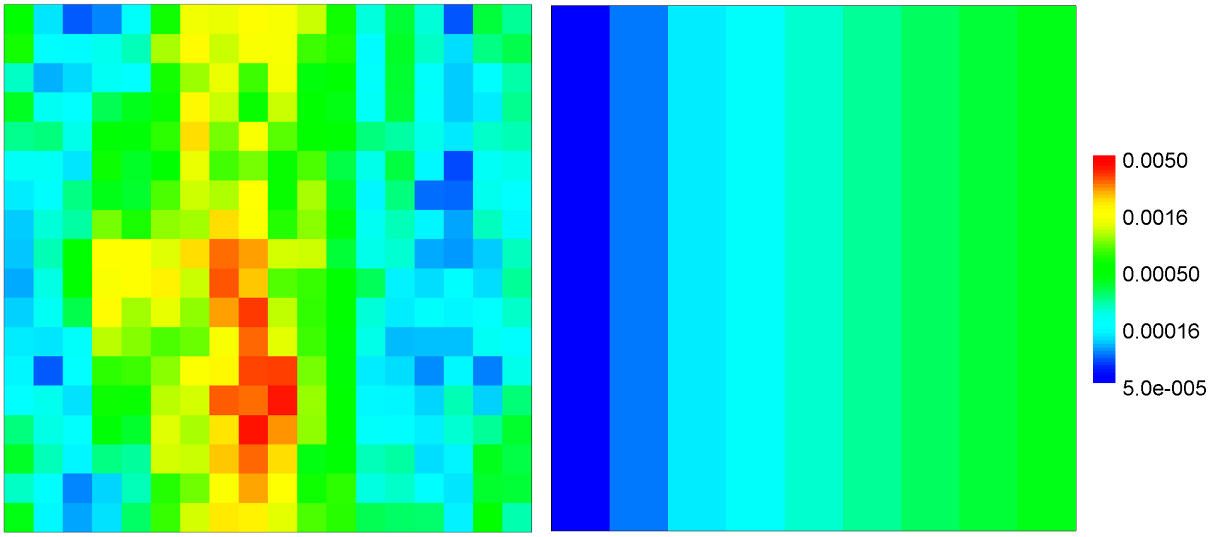
Epilogue: The True System

For this synthetic model, the true system properties and dynamics are known. The representation of all model parameters in the true system is the same as in models A and B, except for the hydraulic conductivity of model layer 1, as shown in Figure 23. For example, the configuration of recharge zones, the thicknesses of model layers, and the location of the river in the true system are the same as shown in Figure 18, and the pattern of increase hydraulic conductivity in layer 2 away from the river is the same in the true system as in models A and B.

The true hydraulic conductivity values in both model layers and all model parameter values are shown in Figure 23. Parameter values for model B are also presented. Model B values do not match true values for four reasons: (1) there was error added to the head and flow data used for calibration, (2) model B could not represent the distribution of hydraulic conductivity in layer 1 correctly, (3) prior information on the parameters was inconsistent with true values, and (4) some parameter values were not well informed by the head and flow observations. The only actual model error is the inability to represent the variation of hydraulic conductivity in model layer 1; the other three reasons for the parameter values being different can be accounted for by the error model, which for this model included the assumption of Gaussian errors with a variances consistent with the observation and prior information weighting. The analysis of weighted residuals for calibrated model B (text following Table 8) correctly indicated model error in model B. Notably, the fit of the calibrated model is still poor overall, as measured by the standard error of 7.6 (Table 8). A value close to 1.0 would have suggested that model fit was consistent with the a priori defined error model.

Comparison of the true parameter values with those estimated for model B shows that the estimate of HK_2 is about an order of magnitude too large, and the recharge estimates are each about 10 cm/yr too large (Figure 23).

In the true system, the drawdown and flow predictions violate the management criteria (Figure 23). In the calibrated model, these predictions barely satisfy the criteria. The uncertainty analysis provides ample warning that this might occur in the true system. Any management decision that uses this uncertainty analysis will explicitly consider the risk involved in drilling the new production well and pumping at a rate of 2.2 m^3/s.



Parameter [[1]Fixed in regression]		True	Estimated, model B
HK_1 (m/s)		*Variable*	
Low K area			2.62×10^{-4}
High K area			1.02×10^{-2}
HK_2 (m/s)		4.4×10^{-5}	3.17×10^{-4}
VANI[1] (---)		1.0	1.0
VK_CB[1] (m/s)		2×10^{-7}	1.0×10^{-7}
K_RB[1] (m/s)		1.0×10^{-3}	1.2×10^{-3}
RCH_1 (cm/yr)		40	50.6
RCH_2 (cm/yr)		70	80.7
Prediction	Criterion	True	Simulated
Drawdown in the southeast corner (m)	< 20	21	17
Head at the well (m)	> 80	134	144
Flow to the stream (m^3/s)	< -4	-3.4	-4.5

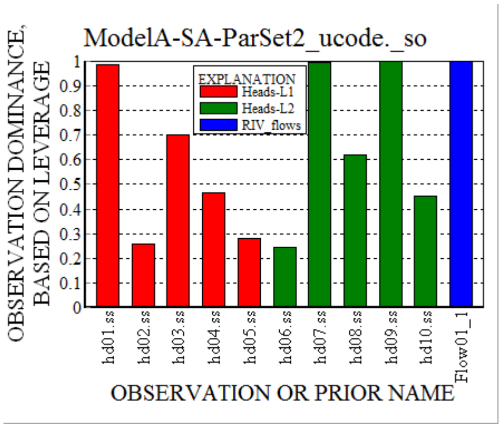
Figure 23. True K field for the upper and lower aquifers (on the left and right, respectively), and true values of estimated parameters and predictions compared to model equivalents. Gray shading indicates predicted value violates management criterion.

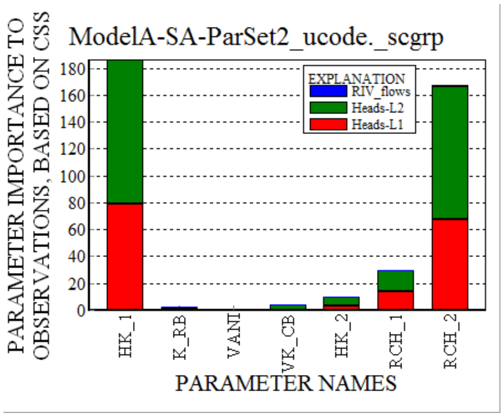
Supplementary Model Analyses

In this section, we mostly present analyses that were considered during and after the model calibration process but were not included in the main report, to simplify the presentation. This is common – rarely are all analyses presented in the report. In certain circumstances some of these could be presented in the main part or an appendix of published reports to emphasize important points or conclusions. Generally, decisions about what to include need to be tailored to the goals of the particular modeling study. Results of identifying important potential new data are also presented in this section; when such analyses are conducted the results are generally included in the main report.

Initial Sensitivity Analysis

In the course of model calibration, graphs such as those in Table 6 are produced many times as initial sensitivity analysis is conducted for different sets of parameter values. Inspecting results at different parameter sets is important because large differences suggest model nonlinearity that could make these analyses unrealiable. Some additional graphs are shown in Table D-3. Here we present two more, showing leverage and CSS calculated using the lower reasonable values from Table 5. In all cases the important and unimportant observations and parameters identified are the same or very similar. This suggests the utility of these computationally frugal analysis methods.

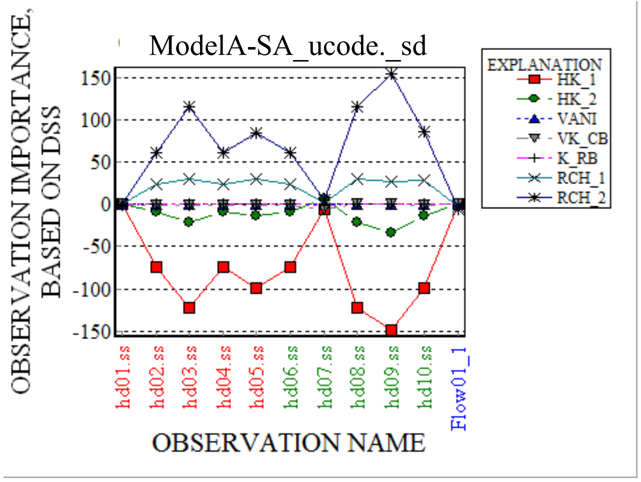
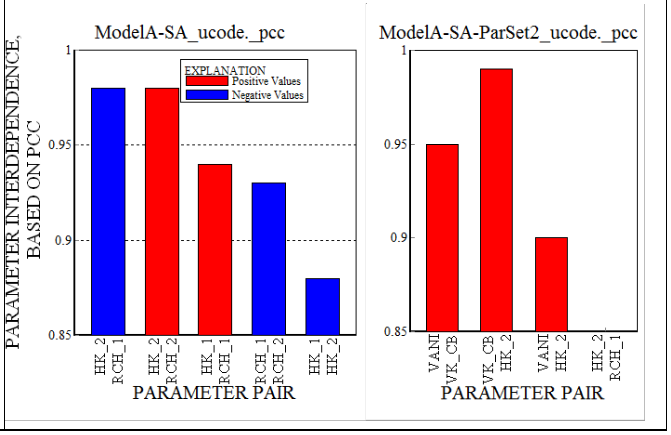




Observations can have high leverage (and high potential for controlling estimated parameter values) and parameters can have high t-statistics (and good prospects of being estimated) because of sensitivity or parameter correlation. An understanding of which of these contribute to the results shown in Table 6 can be obtained using analyses such as those shown in Table 11. Understanding the roles of parameter insensitivity and correlation is useful to understanding simulated system dynamics (and judging whether they are reasonable) and determining how to obtain estimates for the defined parameters. For example, if a parameter is insensitive, the parameter might need to be set to a fixed value or might need prior information. If a parameter is correlated with one or more other parameters, often only one of the parameters needs to be set or have prior information added.

The DSS (Table 11) show that the streamflow gain observation dominance does not stem from sensitivity (note the small DSS values for Flow01_1; this is also reflected in the small contribution of the flow observation to the stacked CSS graphs of Table 6). This means that it is dominant because the flow is important to reducing parameter correlation, which in turn means that the flow dominates the estimation of more than one parameter. Indeed, a sensitivity analysis run conducted without the flow observation shows extreme parameter correlation (PCC=1.00) for all of the parameters except VANI (Hill and Østerby, 2003). This suggests that any error in the flow observation will affect many of the estimated parameter values, and emphasizes the likely importance of any opportunity to reduce the error of this observation or obtain additional streamflow gain or loss observations under different flow conditions so that one observation is not so dominant. Similarly, comparison of the leverage (Table 6) and DSS (Table 11) results shows that the dominance of hd01.ss and hd07.ss is likely because of their roles in reducing parameter correlations.

Table 11. For model A, the role of parameter sensitivity and correlation as measured by dimensionless scaled sensitivities (DSS) and parameter correlation coefficients (PCC).

Question	Graph[1]
Question: Is an observation rated as important using leverage (Table 6) because it contributes to parameter sensitivity, contributes to reducing parameter correlation (thus improving parameter uniqueness), or both? **Answer:** Large leverage, large DSS for at least one parameter: observation contributes to parameter sensitivity and maybe parameter uniqueness. hd05.ss and hd09.ss contribute to parameter sensitivity. **Action:** Large leverage, small DSS for all parameters: observation contributes to uniqueness. hd01.ss, hd07.ss, and Flow01_1 contribute to parameter uniqueness.	Dimensionless scaled sensitivity (DSS) 
Question: What parameters are most dependent (cannot be estimated uniquely with given data)? **Answer:** Results from two sets of parameters indicate different interrelations. This suggests no pervasive dependence given the observations used. Hill and Østerby (2003) show that for a similar model using heads alone, there is pervasive extreme dependence for all parameters except VANI. **Action:** Information on a highly correlated parameter can be obtained through measurements of that parameter or through collection of new observation data that reduce the correlation. For a pair of highly correlated parameters, consider fixing the parameter value that is least uncertain on the basis of independent information.	Parameter correlation coefficient (PCC) calculated at values from Table 5: Starting values Lower reasonable values 

[1] Equations for the plotted quantities are provided in the section “Definition of Statistics, Including Equations”.

Sensitivity analysis can also be used to detect model input errors. For example, observations mistakenly placed in cells with constant-head boundaries can be identified because all leverage and DSS values for such observations will equal zero.

Analysis of Calibrated Model Fit and Parameter Values

Dimensionless scaled sensitivities (DSS) and DFBETAS calculated at the optimal parameter values are shown in Figure 24. These measures can be used together to identify observations that reduce parameter correlation because DSS does not account for parameter correlations, while DFBETAS does account for them. The DFBETAS statistics (Figure 24b) identify four important quantities (in that DFBETAS exceeds the critical value (Hill and Tiedeman (2007), p. 136) for at least one parameter): hd05.ss, hd07.ss, hd09.ss, and Flow01_1. Of these observations, hd07.ss and Flow01_1 have no large DSS values, indicating that these observation are important because they prevent large parameter correlation.

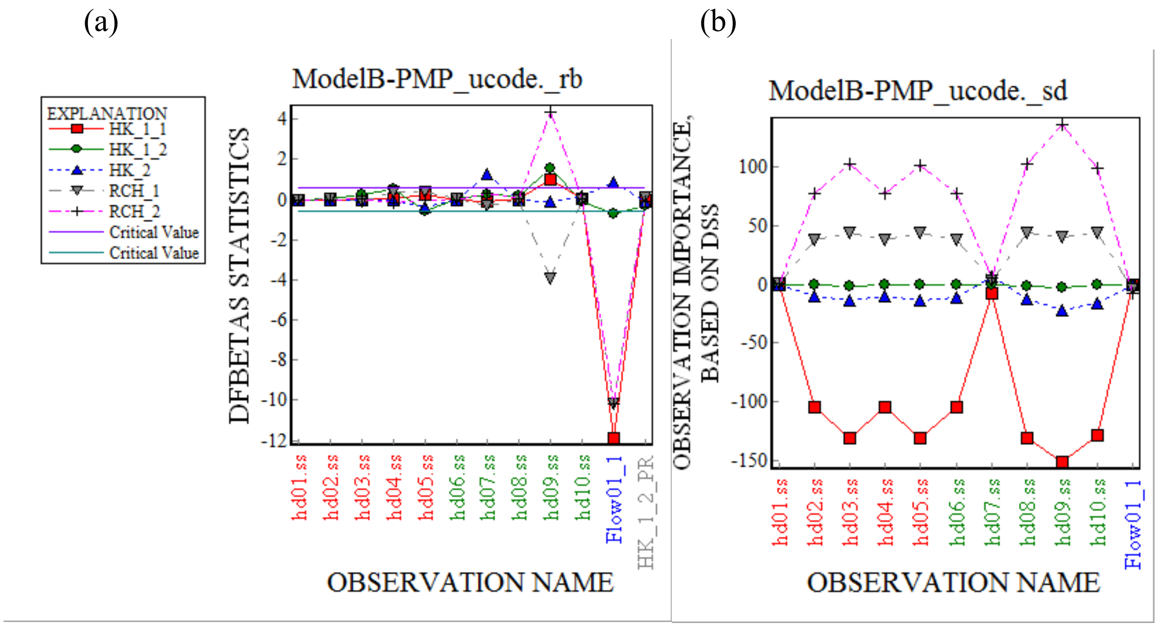


Figure 24. Identifying important observations and prior information for calibrated model B, using (a) dimensionless scaled sensitivities (DSS) and (b) DFBETAS. DSS does not account for parameter correlation or prior information, whereas DFBETAS includes the effects of both correlation and prior.

Analyses for identifying parameter importance are shown in Figure 25. The effect of the estimated parameters on the observation-only objective function is shown in Figure 25a using a basic sensitivity analysis. Most of the parameters are varied by dividing and multiplying by a factor of two, while vertical anisotropy values of 0.8 and 1.2 are considered. Parameter importance measured by CSS is shown in Figure 25b. CSS is calculated by any UCODE sensitivity analysis or regression run. The CSS and the basic sensitivity analysis produce identical rankings for the four most important parameters. In order of decreasing importance, the parameter rankings are: HK_1_1, RCH_2, RCH_1, HK_2.

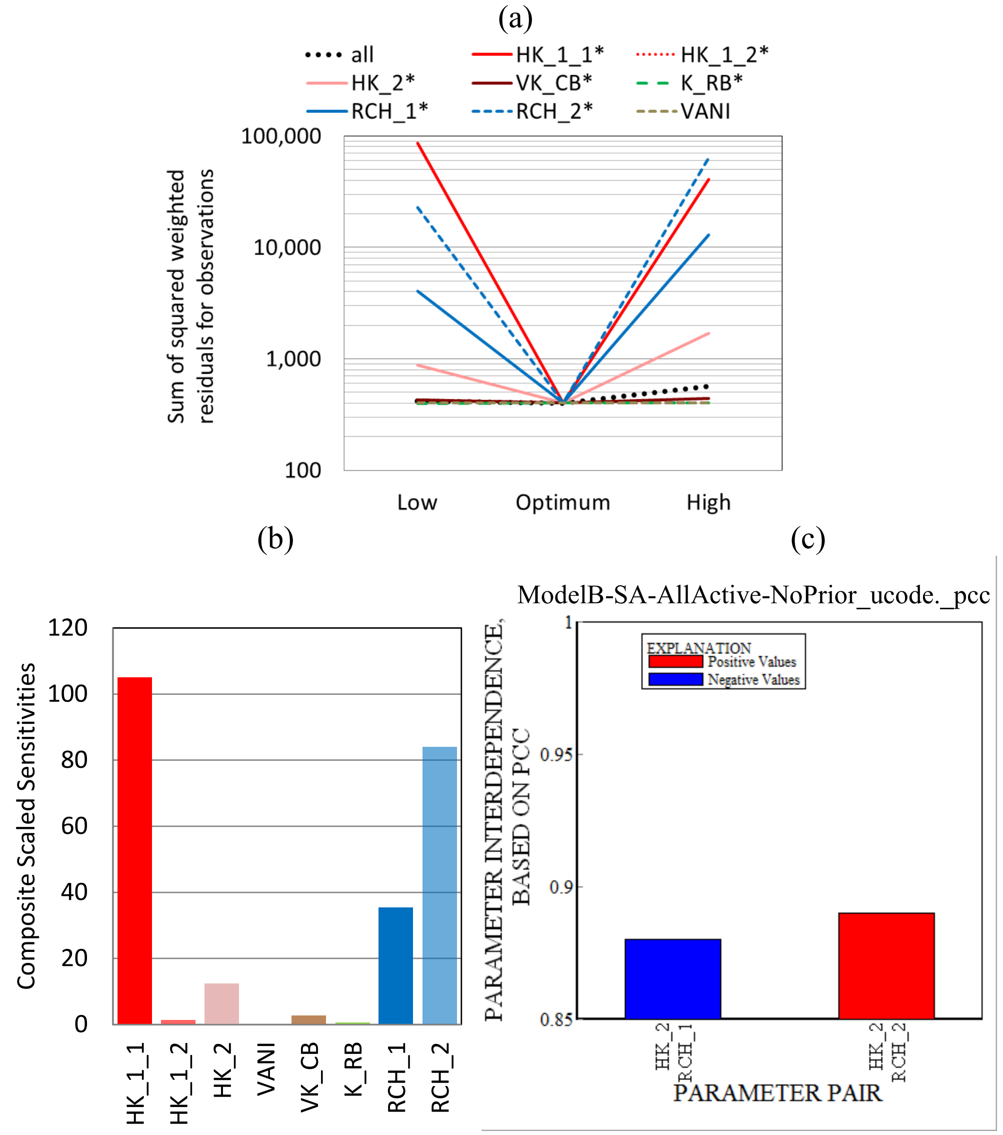


Figure 25. The importance of all defined parameters to observations only (without prior information) calculated at model B optimal parameter estimates. (a) Objective-function values produced by changing parameter values from the optimal estimates. Most parameters (marked by * in the legend) are changed by dividing and multiplying by 2. VANI has low and high values of 0.8 and 1.2. (b) Composite scaled sensitivities (CSS) calculated with all parameters activated. The bars are colored to match the line colors in (a) so that results can easily be compared for each parameter. (c) Parameter correlation coefficients (PCC). This graph shows only parameter pairs with PCC magnitudes ≥ 0.85. Values close to 1.0 indicate parameter dependence.

The basic sensitivity analysis (Figure 25a) shows that many of the parameter sets produce very poor model fits. The objective function value measuring fit to observations increases to >10,000 for several of the parameter changes, while the value at the optimal parameter estimates is 401 ($S(\boldsymbol{b})_{obs}$ in Table 8). Values of the objective function are dimensionless, but gain meaning by noting that the objective function would be 7.0 if the model fit were consistent with the weighting used. This is because the value of the calculated error variance, s^2, would be 1.0 in this circumstance, and $s^2=S/(nh+nq+npr-np)$ where *np* is the number of parameters. Therefore the large objective function values produced by the parameter changes shown in Figure 25a clearly indicate that some unrealistic parameter values are used in the basic sensitivity analysis. It also raises the issue of how large is too large, with respect to the objective function produced by this sensitivity analysis. To address this issue, consider the objective function value used to calculate nonlinear confidence intervals. This can be calculated using equation 8.14 of Hill and Tiedeman (2007, p. 178) as

$$S(\boldsymbol{b}) + s^2 \times critical\ value \qquad (14)$$

For a 95% individual confidence intervals, the critical value equals about 2.0. For calibrated model B, $S(\boldsymbol{b})$=405 (Table 8) and $S(\boldsymbol{b})/7=s^2=58$. Multiplying by a critical value of 2 gives a value for equation 14 of 521. Values larger than 521 can be thought of as very large. In Figure 25a the objective function for the four most important parameters (HK_1_1, HK_2, RCH_1, and RCH_2) clearly exceeds this threshold, suggesting that the parameter values used in the analysis are unrealistic.

Figure 25c shows that there are no parameter pairs that are highly correlated, even with all parameters activated and no prior information (PCC for the conditions of the model calibration run are smaller and are shown in Appendix D, Table D-3). While numerical issues may sometimes cause calculated PCC to be too small (Hill and Tiedeman, 2007, p. 58-59), it is known for this inverse model that the presence of the flow observation prevents the parameters from being extremely correlated, as discussed earlier. The results in Figure 25c also show how PCC can dramatically vary when calculated at different parameter values (also illustrated by Hill and Tiedeman, 2007, p. 58, fig. 4.2). These results are very different from those calculated for model A at two different sets of parameter values and with no prior (Table 11). The results also indicate that splitting HK_1 into two parameters for model B does not increase the dependence.

Prediction Uncertainty

In this work prediction uncertainty such as displayed in Figure 22 is quantified by the common approach of propagating the uncertainty of the parameters to the predictions. Here, the propagation is accomplished using linear inferential theory, which includes some assumptions not required by methods such as MCMC, as discussed in Chapter 5 of this documentation. Briefly, linear inference depends on the model-simulated predictions being approximately linear functions of the parameters within the range of parameter uncertainty considered. Further, it depends on the errors being approximately Gaussian. Lu et al. (2012) discuss and test these assumptions.

Understanding sources of prediction uncertainty thus requires consideration of the parameter uncertainty and the dependence of the predictions on the different parameters. For methods such as MCMC this is equivalent to the requirements of (i) considering the range of parameter values to define and (ii) calculating the prediction with different sets of parameter values. Generally, if a prediction is highly dependent on a parameter that is uncertain, prediction uncertainty will be relatively large.

Parameter uncertainty can be investigated using measures of information that the observations provide on the parameters, as discussed for Figure 25. Parameters with more information provided by observations tend to be less uncertain. Parameter uncertainty can be measured more quantitatively using confidence intervals such as those shown in Table 7 and in Figure 26a-b. The 95% linear individual confidence intervals shown are a direct, though not complete, measure of the uncertainty propagated to prediction uncertainty using linear inference. These parameter confidence intervals are calculated with the variances of the parameter variance-covariance matrix. The linear prediction confidence intervals are calculated with the parameter variances as well as their covariances.

The confidence intervals in Figure 26a-b differ from those in Table 7 in that they are calculated with all parameters activated and prior information on all parameters. These calculation conditions are important when propagating parameter uncertainty to prediction uncertainty, so that the propagation includes all model parameters, and includes parameter support provided by the independent information and by the observations.

The dependence of predictions on parameters is examined in two ways in Figure 26. Figure 26c shows prediction scaled sensitivities for which large absolute values identify parameters important to the predictions. Figures 26d-f show a basic sensitivity analysis that identifies the same parameters as being important to the predictions. Figures 26c-f suggest that the predictions are dependent on HK_1_1 and the two recharge parameters, but depend very little on parameters HK_1_2, HK_2, VANI, K_RB, and VK_CB.

Readers familiar with Bayesian theory will wonder why the a posteriori parameter confidence intervals of Figure 26a-b are larger than the reasonable ranges. The reasonable ranges are 95% confidence intervals which are consistent with the prior information used for the uncertainty analysis of model B. Typically, Bayes theory allows the observations to reduce the uncertainty of parameters from that defined using the prior information. Yet here, the uncertainty has increased.

The reason for the increase is that the methods used in this work treat prior information differently than in Bayesian methods. Instead of being treated as known, it is treated as a hypothesis. If simulated results do not match the observations very well and the fit requires parameters to deviate from the prior values significantly, the prior information is considered potentially less applicable to the model as constructed than was originally thought. A regression standard error of 1.0 would have indicated that preconceived notions of model accuracy and observation and prior accuracy were consistent. In this work the standard error of the regression is 7.6 (Table 8), indicating inadequacies in the flow model, in the error model used to quantify uncertainty in the observations and prior information, or all of these. As a result, the methods used increase the uncertainty of the prior information. The result is that the a posteriori confidence intervals are larger than the a priori confidence intervals for this problem.

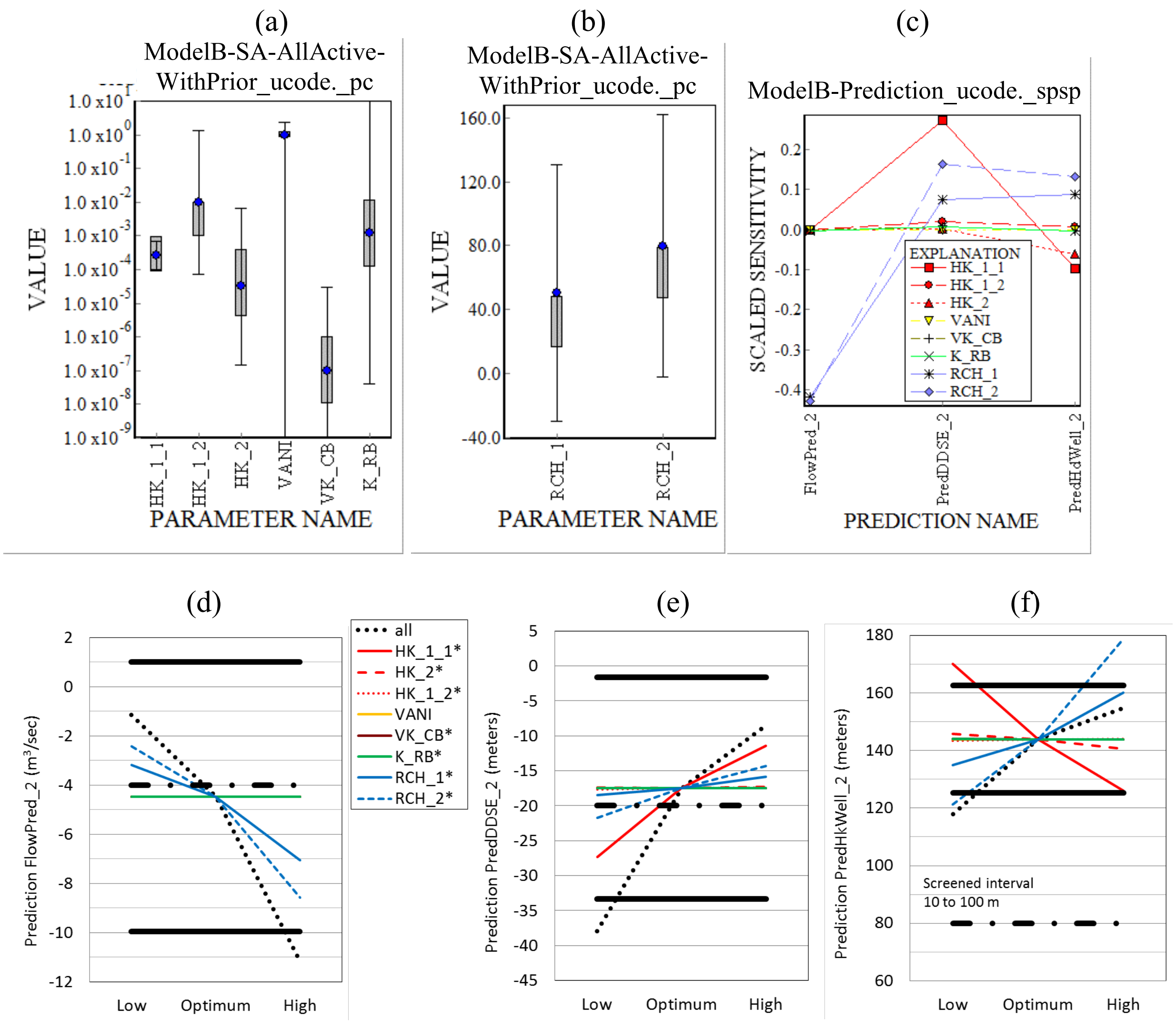


Figure 26. Investigating the sources of prediction uncertainty. (a and b) Parameter estimates (blue dots), 95% linear individual confidence intervals (black lines) calculated with prior information on all parameters, and reasonable ranges (gray bars). The uncertainty quantified in the parameter confidence intervals is propagated to calculate prediction confidence intervals. This propagation is examined in (c-f). (c) Parameter importance to predictions as measured by prediction sensitivities scaled by the calculated parameter standard deviation divided by the prediction value. If the absolute value of the scaled sensitivity is large and the parameter confidence interval in (a) or (b) is large, that parameter significantly contributes to the width of the prediction confidence interval. (d, e, and f) An alternate way of measuring parameter importance to predictions, showing changes for each prediction produced by changing the parameter values one at a time or all together. Most parameters (marked by * in the legend) are changed by dividing and multiplying by 2. The management criteria for the predictions are marked by a dot-dash line, and confidence interval limits are the solid black lines. In (d) and (f), the line for K_RB covers the lines for several other parameters. Panels (a) and (b) suggest that for some parameters the factor of two change is too small.

Using the Model to Evaluate Potential New Data Collection

In some circumstances modelers are asked to evaluate what new data might be useful for improving the predictions. This section provides an example of how this question could be addressed for the flow system and predictions considered. It includes evaluation of (i) potential new head and flow observations and (ii) potential new independent information about flow system properties related to the model parameters. These analyses are conducted using the OPR-PPR software (Tonkin et al., 2007), briefly described in Chapter 6.

Potential New Observations

A cost effective approach for obtaining new head observations is to collect them at the existing monitoring wells under flow system conditions that involve a change in hydraulic stress compared to the conditions for which steady-state model B was calibrated. The predictive scenario includes a new pumping well. A scenario that might yield information valuable to the predictions is to collect new observations under conditions of pumping from one of the existing monitoring wells. Data collection under these conditions would be an efficient method of gaining information about the effects of pumping before committing to the expense of a new production well. Well 03 in the upper aquifer and well 08 in the lower aquifer (Figure 19) were selected for this analysis because they are each close to the proposed location of the new production well. Because of well construction considerations and because each monitor well is open to only one aquifer, unlike the proposed multi-aquifer production well, the sustainable steady-state pumping rate in these monitor wells is likely be lower than the proposed production well pumping rate of 2.2 m^3/s. Thus, for evaluating the value of new field data, a rate of 1.0 m^3/s is simulated for well 03, and a rate of 0.5 m^3/s is simulated for well 08.

For the condition of steady-state pumping in well 03, potential observations include heads in all monitor wells except well 03, and discharge along the entire river reach. Similarly, for the steady-state pumping in well 08, potential observations include heads in all wells except well 08, and discharge to the river. The analysis uses the OPR (observation-prediction) statistic (Hill and Tiedeman, 2007, p. 171-174; Tonkin et al., 2007) to assess the reduction in prediction uncertainty that would occur by adding the potential observations. The statistics are calculated for the cases of (1) adding the observations individually (Figure 27a), and (2) adding the observations in two groups, one group corresponding to all observations collected during pumping of well 03, and the other corresponding to all observations during pumping of well 08 (Figure 27b).

Results show that a given observation or group tends to have a similar degree of importance to all three predictions, and that head and flow observations for the scenario of pumping monitor well 03 are more important to the predictions than are the observations for the scenario of pumping well 08 (Figure 27). The results also show that it is not necessary to monitor heads at wells 01 or 07 during either of these scenarios, as these observations are unimportant to the predictions.

The predictions are most sensitive to HK_1_1 and the two recharge parameters (Figure 26c). Sensitivities for the potential observations to the recharge parameters (not shown) tend to be slightly larger for the case of pumping well 03 at 1 m^3/s than for the case of pumping well 08 at

0.5 m^3/s (if both aquifers were confined, these sensitivities would be identical; however, the system is simulated as unconfined). These larger recharge sensitivities appear to be a main factor contributing to the result that observations potentially collected during pumping of well 03 rank as more important than those potentially collected during pumping of well 08.

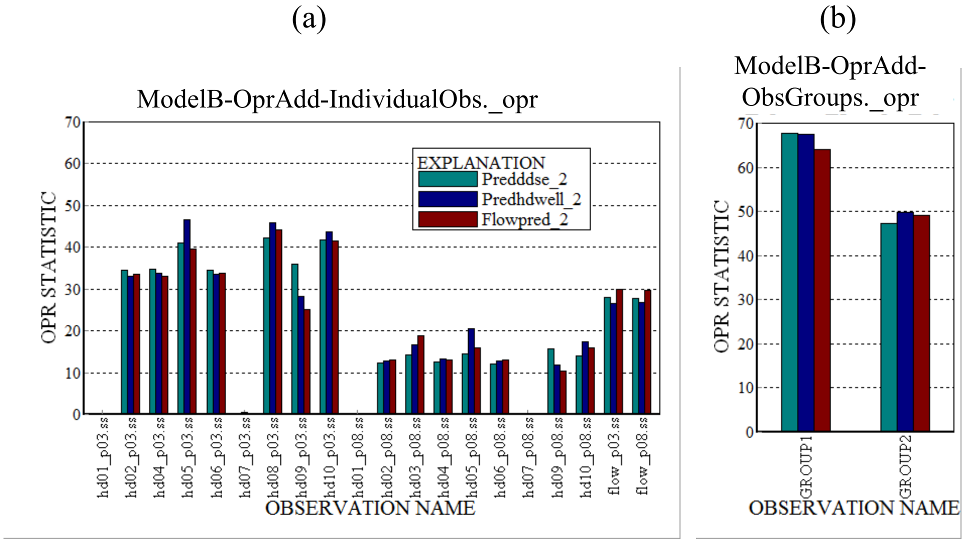


Figure 27. The importance of potential new observations to each prediction, as measured by the OPR (observation-prediction) statistic. Here, OPR measures the percent decrease in the prediction standard deviation produced by adding (a) an individual observation or (b) a group of observations. Group 1 includes all head and flow observations potentially collected during pumping of well 03, and Group 2 includes all heads and flows potentially collected during pumping of well 08.

The OPR statistic is defined as a percent decrease in prediction uncertainty. However, best practice for this statistic involves using it to rank potential observation importance in a relative manner. After any new observation data are collected, these data will most likely be used, along with the existing observations, to recalibrate the model. When the recalibrated model is used to update the predictions and their uncertainty, it is likely that the reduction in uncertainty will differ from the percentages shown in Figure 27.

Potential New System Information

The PPR (parameter-prediction) statistic (Hill and Tiedeman, 2007, p. 166-170; Tonkin et al., 2007) is used to evaluate the importance to the predictions of collecting additional field data about flow system properties related to the model parameters. The analysis assumes that independent information about a property is collected that results in a specified percent reduction in the uncertainty of the prior information on the associated parameter. Here, a 10 percent reduction in uncertainty is used.

Results show that among the eight flow system parameters, collecting new information about HK_1_1 would be most beneficial to reducing prediction uncertainty (Figure 28). Collecting recharge information close to the hillside also would be quite beneficial for reducing uncertainty in predicted head in the pumping well. However, the prediction uncertainty analysis showed that the uncertainty in the flow and drawdown predictions were the most problematic, in terms of assessing whether the proposed production-well pumping would violate the management criteria. Given this, the PPR results suggest that hydrogeologic field work would best be focused on better characterizing the area of the upper aquifer where HK_1_1 is defined. In practice, this additional characterization could then be used to refine the representation of the hydraulic conductivity distribution in the upper aquifer, improve the prior value for HK_1_1, and (or) reduce the uncertainty on this prior (i.e., reduce the standard deviation used to calculate its weight).

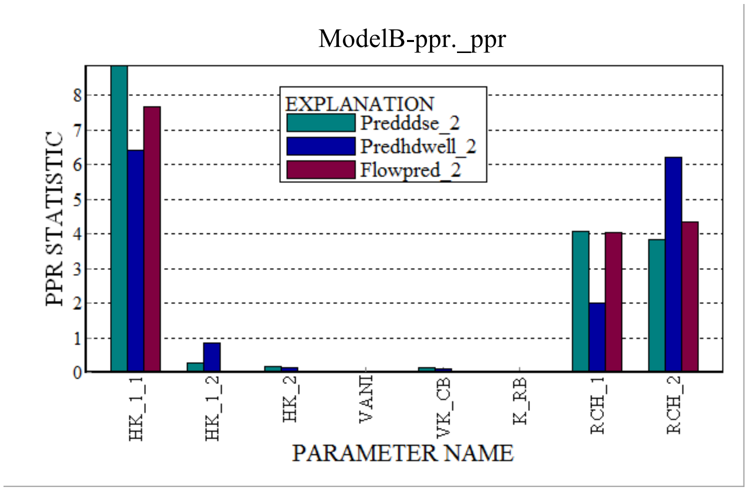


Figure 28. The importance of model parameters to the predictions, as measured by the PPR (parameter-prediction) statistic. Here, PPR measures the percent decrease in a prediction standard deviation produced by a ten-percent reduction in the uncertainty of the prior information on each parameter.

Definition of Statistics, Including Equations and Comments on Meaning and Use

$$\text{Cook's D} = \frac{(\underline{b}_{(i)} - \underline{b}')^{\mathrm{T}} \left[\sigma^2 (\mathbf{X}^T \boldsymbol{\omega} \mathbf{X})^{-1}\right]^{-1} (\underline{b}_{(i)} - \underline{b}')}{N\mathrm{P}} = \frac{1}{NP} r_i^2 \frac{h_{ii}}{1\text{-}h_{ii}}$$

where

$\underline{b}'$ is the set of parameter values optimized using all observations;
$\underline{b}_{(i)}$ is the set of parameter values that would be estimated if just the i^{th} observation is omitted;
$\boldsymbol{X}$ is a matrix of sensitivities (see Hill and Tiedeman, 2007, p. 69);
$\boldsymbol{\omega}$ is the weight matrix of equation 11 augmented with the prior information weight matrix (see Hill and Tiedeman, 2007, p. 384);
NP is the number of estimated parameters;
σ^2 is the variance of the regression;
r_i is the i^{th} weighted residual divided by its standard error, calculated as $f_i/[\sigma(1\text{-}h_{ii})^{1/2}]$;
f_i is the i^{th} weighted residual of the regression with all observations; and
h_{ii} is the leverage of the i^{th} observation.

Observations with Cook's D greater than the critical value strongly control one or more of the estimated parameter values. Cook's D includes the effects of model fit, so is often said to be a measure of actual importance, while leverage is said to be a measure of potential importance. Often the parameters shown to be important by Cook's D are a subset of the observations shown to be potentially important by leverage. Cook's D includes the effects of parameter correlation.

$CSS_j = ((\sum_{i=1,nobs} DSS_{ij}^2)/nobs)^{1/2}$, where *nobs* is the number of observations. CSS stands for composite scaled sensitivity. Measures parameter importance to observations, and thus how well the parameter is informed by observations. Larger values indicate more important parameters. Does not include effects of parameter intereactions (correlation). Can be stacked to show observation contribution to each parameter.

$DSS_{ij} = \omega_i^{½} b_j (\partial y_i/\partial b_j)$, where ω_i is the weight for observation *i*, b_j is the value of parameter *j*, and $\partial y_i/\partial b_j$ is the derivative of the simulated equivalent of observation y_i with respect to parameter b_j. DSS stands for dimensionless scaled sensitivity. Larger absolute values indicate more information is provided by the observation for the parameter. See Hill and Tiedeman (2007) for discussion of issues such as what happens when b_j is zero, the weight matrix is full, and so on.

Leverage = $\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{\omega}\boldsymbol{X})^{-1}\boldsymbol{x}_i$. , where $\boldsymbol{X}$ is a sensitivity matrix with elements equal to $\partial y_i/\partial b_j$, and x_i is a row of $\boldsymbol{X}$. $\boldsymbol{\omega}$ is defined after equation 11. Leverage includes the effects of parameter correlation. Leverage is fit-independent in that model fit for observations is not included in its calculation. Values range from 0.0 to 1.0. Observations with leverage close to 1.0 can control one or more of the estimated parameter values.

$PCC_{jk} = V_{jk}/(V_{jj}^{½}V_{kk}^{½})$, where $V_{jk}=(\boldsymbol{X}^T\boldsymbol{\omega}\boldsymbol{X})^{-1}_{jk}$. PCC stands for parameter correlation coefficient. Values vary from -1.0 to 1.0. Absolute values near 1.0 indicate high parameter correlation.

t-statistic, scaled to be fit-independent, equals the parameter value divided by the fit-independent version of the parameter standard deviation. For parameter *j*, the parameter standard deviation is calculated as $[s^2(\boldsymbol{X}^T\boldsymbol{\omega}\boldsymbol{X})^{-1}_{jj}]^{1/2}$. It is made fit-independent by dividing the quantity in brackets by s^2. In this work, these values were calculated with all parameters being native. That is, none are log-transformed. Combining results for transformed and native parameters can produce results that do not clearly identify important and unimportant parameters.

Chapter 10: REFERENCES

Anderman, E.R., Hill, M.C. and Poeter, E.P., 1996, Two-dimensional advective transport in ground-water flow parameter estimation: Ground Water, 34(6):1001-1009.

Anderman, E.R. and Hill, M.C., 1999, A new multistage groundwater transport inverse method: Presentation, evaluation, and implications: Water Resources Research., v. 35, No. 4, p. 1053-1063.

Aster, R.C., Borchers, B., and Thurber, C.H., 2012, Parameter estimation and inverse problems, second edition: Elsevier Academic Press, Burlington, Massachusetts, 360pp.

Barlebo, H.C., Hill, M.C., Rosbjerg, D., and Jensen, K.H., 1998, Concentration data and dimensionality in groundwater models: Evaluation using inverse modeling: Nordic Hydrology, v. 29, p. 149-178.

Banta, E.R., 2011, ModelMate—A Graphical User Interface for Model Analysis: U.S. Geological Survey Techniques and Methods, book 6, chap. E4, 31 p. http://water.usgs.gov/software/ModelMate/

Banta, E.R., Poeter, E.P., Doherty, J.E., Hill, M.C., 2006, JUPITER API: U.S. Geological Survey Techniques and Methods 6-E1, 268p. URL: http://water.usgs.gov/nrp/gwsoftware/jupiter/jupiter_api.html

Barlebo, H.C., Hill, M.C., Rosbjerg, Dan, and Jensen, K.H., 1998, Concentration data and dimensionality in groundwater models, Evaluation using inverse modelling: Nordic Hydrology, v. 29, p. 149-178.

Bates, D.M. and D.G. Watts, 1980, Relative curvature measures in nonlinearity: Journal of the Royal Statistical Society, Series B, vol. 42, no. 1, p. 1-25.

Borgonovo, E., Casaings, W., Tarantola, S., 2012, Model emulation and moment-independent sensitivity analysis: An application to environmental modelling: Environmental Modelling and Software, 34:105-115.

Borgonovo, E. and Smith, C.L., 2011, A study in the risk assessment of complex engineering systems: An application to space PSA. Operations Research 59(6):1461-1476.

Box, E.P, G.C. Tiao, 1992, Bayesian Inference in Statistical Analysis (Wiley, New York [Original work published 1973], doi:10.1002/9781118033197.

Carter, R.W and Anderson, I.E, 1963, Accuracy of current meter measurements: American Society of Engineers Journal, 89(HV4):105-115.

Christensen, S., and R. L. Cooley, 2005, User guide to the UNC process and three utility programs for computation of nonlinear confidence and prediction intervals using MODFLOW-2000, U.S. Geological Survey Techniques and Methods Report 2004-1349.

Cooley, R. L., 2004, A theory for modeling groundwater flow in heterogeneous media, U. S. Geological Survey Professional Paper 1979.

Dausman, A.M., Doherty, J., Langevin, C.D., and Sukop, M.C., (2009). Quantifying Data Worth Toward Reducing Predictive Uncertainty: Ground Water, 48(5):729–740.

Dennis, J.E., Gay, D.M., and Welch, R.E, 1981, An adaptive nonlinear least-squares algorithm: ACM Transactions on Mathematical Software, 7(3):338-368.

Doherty, J., 2010, PEST, Model-independent parameter estimation—User manual (5th ed., with slight additions) and addendum: Brisbane, Australia, Watermark Numerical Computing. http://www.sspa.com/PEST/index.html

Doherty, J., R.J. Hunt, 2009, Two statistics for evaluating parameter identifiability and error reduction. Journal of Hydrology 366, 119–127.

Doherty. J., R.J. Hunt, 2010a, Response to Comment on ‘‘Two statistics for evaluating parameter identifiability and error reduction”: Journal of Hydrology 380: 489–496.

Doherty, J.E., and Hunt, R.J., 2010b, Approaches to highly parameterized inversion—A guide to using PEST for groundwater-model calibration: U.S. Geological Survey Scientific Investigations Report 2010–5169, 59 p.

Doherty, J., and D. Welter, 2010, A short exploration of structural noise, Water Resour. Res., 46, W05525, doi:10.1029/2009WR008377.

Fienen, M. N., Muffels, C. T., and Hunt, R. J., 2009, On constraining pilot point calibration with regularization in PEST: Ground Water, v. 47, issue 6, p. 835-844. doi:10. 1111/j. 1745-6584. 2009. 00579. X

Fienen, M.N., Doherty, J.E., Hunt, R.J., and Reeves, H.W., 2010, Using Prediction Uncertainty Analysis to Design Hydrologic Monitoring Networks: Example Applications from the Great Lakes Water Availability Pilot Project: U.S. Geological Survey Scientific Investigations Report 2010–5159, 44 p. http://pubs.usgs.gov/sir/2010/5159

Fienen, M.N., D’Oria, Marco, Doherty, J.E., and Hunt, R.J., 2013, Approaches in highly parameterized inversion: bgaPEST, a Bayesian geostatistical approach implementation with PEST—Documentation and instructions: U.S. Geological Survey Techniques and Methods, book 7, section C9 , 86 p. Documentation available at http://pubs.usgs.gov/tm/07/c09. Program available at https://github.com/mnfienen/bgaPEST.

Finsterle, S. and Kowalsky, M.B., 2011, A truncated Levenberg-Marquardt algorithm for the calibration of highly parameterized nonlinear models, Computers and Geoscience, 37:731-738.

Foglia, L., M.C. Hill, S. W. Mehl, and P. Burlando, 2009, Sensitivity analysis, calibration, and testing of a distributed hydrological model using error-based weighting and one objective function, Water Resour. Res., 45, W06427, doi:10.1029/2008WR007255.

Foglia, L., S. W. Mehl, M. C. Hill, and P. Burlando, 2013, Evaluating model structure adequacy: The case of the Maggia Valley groundwater system, southern Switzerland, Water Resour. Res., 49, doi:10.1029/2011WR011779.

Gallagher, M., and J. Doherty, 2007, Parameter estimation and uncertainty analysis for a watershed model, Environ. Model. Softw. 22, 1000–1020, doi:10.1016/j.envsoft.2006.06.007.

Gelman, A., Rubin, D.B., 1992, Inference from iterative simulation using multiple sequences. Statistical science, 7(4), 457-472.

Golub, G.H. and Van Loan, C.F., 1996, Matrix Computations, 3. Ed. Johns Hopkins.Haber, Eldad, Uri M Ascher, and Doug Oldenburg, 2000, On optimization techniques for solving nonlinear inverse problems: Inverse Problems, 16 1263, doi: 10.1088/0266-5611/16/5/309

Gupta, H. V., M. P. Clark, J. A. Vrugt, G. Abramowitz, and M. Ye (2012), Towards a comprehensive assessment of model structural adequacy, Water Resour. Res., 48, W08301, doi:10.1029/2011WR011044.

Hanson, R.T., Kauffman, L.K., Hill, M.C., Dickinson, J.E., and Mehl, S.W., 2013, Advective transport observations with MODPATH-OBS—Documentation of the MODPATH observation process, using four types of observations and predictions: U.S. Geological Survey Techniques and Methods book 6–chap. A42, 96 p., http://pubs.usgs.gov/tm/06/a42 .

Harbaugh, A.W., 2005, MODFLOW-2005, The U.S. Geological Survey modular ground-water model—the Ground-Water Flow Process: U.S. Geological Survey Techniques and Methods 6-A16, variously p.

Hill, M.C., 1992, A computer program (MODFLOWP) for estimating parameters of a transient, three- dimensional, groundwater flow model using nonlinear regression, U.S. Geological Survey Open-File Report 91-484 pending publication in Techniques of Water-Resources Investigations, 358 p.

Hill, M.C., 1998, Methods and guidelines for effective model calibration: U.S Geological Survey Water- Resources Investigations Report 98-4005, 90p. http://pubs.water.usgs.gov/wri984005/

Hill, M.C., 2008, Data error and highly parameterized groundwater models, in eds. Jens Christian Refsgaard and others, Calibration and reliability in groundwater modeling, Credibility of Modeling, Proceeding of the ModelCARE2007 Conference, Copenhagen, Denmark: IAHS Publication 320, p. 316-321

Hill, M.C, 2010, Comment on ‘‘Two statistics for evaluating parameter identifiability and error reduction" by John Doherty and Randall J. Hunt: Journal of Hydrology 380: 481-488.

Hill, M.C., Banta, E.R., Harbaugh, A.W., and Anderman, E.R., 2000, MODFLOW-2000, the U.S. Geological Survey modular groundwater model, User's guide to the Observation, Sensitivity, and Parameter-Estimation Process and three post-processing programs: U.S. Geological Survey Open-File Report 00-184, 209 p. http://water.usgs.gov/nrp/gwsoftware/modflow2000/modflow2000.html

Hill, M.C. and Nolan, B.T., 2011, Sensitivity analysis for inverse problems solved by singular value decomposition (SVD) in eds Poeter, E.P, Hill, M.C., Zheng, C. and Maxwell, R., Proceedings of the 2011 MODFLOW and More Conference, Golden, Colorado, June 2011.

Hill, M.C. and Tiedeman, C.R., 2007, Effective groundwater model calibration, with analysis of sensitivities, predictions, and uncertainty: Wiley and Sons, New York, New York, 455 p.

Hill, M.C. and Østerby, Ole, 2003, Determining extreme parameter correlation in ground-water models: Ground Water, v. 41, No. 4, p. 420-430.

Hsieh, P.A., and Winston, R.B., 2002, User's Guide to Model Viewer, A Program For Three-Dimensional Visualization of Ground-water Model Results: U.S. Geological Survey Open-File Report 02-106, 18 p.

Jacobson, E.A., 1985, A statistical parameter estimation method using singular value decomposition with application to Avra valley aquifer in southern Arizona: unpublished PhD dissertation, Tucson, University of Arizona, Department of Hydrology and Water Resources.

Kavetski D, Clark MP. 2010. Ancient numerical daemons of conceptual hydrological modeling. Part 2: Impact of time stepping schemes on model analysis and prediction. Water Resources Research 46: W10511, DOI: 10.1029/2009WR008896.

Kavetski, D., and G. Kuczera, 2007, Model smoothing strategies to remove microscale discontinuities and spurious secondary optima in objective functions in hydrological calibration, Water Resour. Res., 43, W03411, doi:10.1029/2006WR005195.

Keidser, A. and Rosbjerg, D., 1991, A comparison of four inverse approaches to groundwater flow and transport parameter identification: Water Resources Research, v. 27, No. 9, p. 2219-2232.

Lu, D., M. Ye, and M. C. Hill, 2012, Analysis of regression confidence intervals and Bayesian credible intervals for uncertainty quantification, Water Resources Research, 48, W09521, doi:10.1029/2011WR011289.

Matott, L. S., 2010, OSTRICH: An Optimization Software Tool. Accessed 12/27/2010 at http://www.civil.uwaterloo.ca/lsmatott/Ostrich/OstrichMain.html

McCabe, G.J., and Markstrom, S.L., 2007, A monthly water-balance model driven by a graphical user interface: U.S. Geological Survey Open-File report 2007-1088, 6 p. http://pubs.usgs.gov/of/2007/1088/pdf/of07-1088_508.pdf

Mehl, S.W. and Hill, M.C., 2010, Grid-size dependence of Cauchy boundary conditions used to simulate stream–aquifer interactions: Advances in Water Resources, 33 (2010) 430–442.

Mroczkowski M., G.P. Raper, G. Kuczera, 1997, The quest for more powerful validation of conceptual catchment models. Water Resources Research, 33(10), 2325-2336.

Oliver D.S., A.C. Reynolds, N. Liu, 2008, Inverse theory for petroleum reservoir characterization and history matching: Cambridge University Press.

Oreskes, N., K. Shrader-Frechette, K. Belitz, 1994, Verification, validation, and confirmation of numerical models in the earth sciences. Science, 263:641-646.

Papoulis, A., 1991, Probability, Random Variables, and Stochastic Processes, 3rd ed., McGraw-Hill, New York.

Plischke, E., E. Borgonovo, and C. L. Smith, 2013, Global sensitivity measures from given data, Eur. J. Oper. Res., 226(3), 536–550, doi:10.1016/j.ejor.2012.11.047.

Poeter, E.P., Hill, M.C., and Banta, E.B., 2005, UCODE_2005 and six other computer codes for universal sensitivity analysis, calibration, and uncertainty evaluation: U.S. Geological Survey Techniques and Methods, book 6, sec. A, chap. 11, 283 p. http://igwmc.mines.edu/freeware/ucode/

Poeter, E.P. and Hill, M.C., 2007, MMA, a computer program for multi-model analysis: U.S Geological Survey Techniques and Methods 6-E3 113p. http://typhoon.mines.edu/freeware/mma/.

Poeter, Eileen P. and M.C. Hill, 2008, SIM_ADJUST – A Computer Code that Adjusts Simulated Equivalents for Observations or Predictions: International Ground Water Modeling Center, Golden, Colorado, Report GWMI 2008-01, 28p. http://igwmc.mines.edu/freeware/sim_adjust/

Rakovec, O., M. C. Hill, M. P. Clark, A. H. Weerts, A. J. Teuling, and R. Uijlenhoet, 2013, Distributed evaluation of local sensitivity analysis (DELSA), with application to hydrologic models, Water Resour. Res., 49, doi:10.1002/2013WR014063.

Renard B., D. Kavetski, E. Leblois, M. Thyer, G. Kuczera, et al. (2011), Toward a reliable decomposition of predictive uncertainty in hydrological modeling: Characterizing rainfall errors using conditional simulation. Water Resour. Res., 47, W11516, doi:10.1029/2011WR010643.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., Tarantola, S., 2008. Global Sensitivity Analysis. The Primer. John Wiley and Sons.

Saltelli, A., P. Annoni, 2010, How to avoid a perfunctory sensitivity analysis, Environmental Modelling & Software 25:1508-1517

Sheets, R.A., M.C. Hill, H.M. Haitjema, A.M. Provost, and J.P. Masterson, 2014, Simulation of water-table aquifers using specified saturated thickness: Ground Water, doi: 10.1111/gwat.12164.

Šimůnek, J., Šejna, M. and van Genuchten, M.T., 1999. The HYDRUS-2D software package for simulating two-dimensional movement of water, heat, and multiple solutes in variably saturated media. Version 2.0. Agricultural Research Service, US Department of Agriculture, Riverside, California.

Šimůnek, J., van Genuchten, M.T. and Šenja, M., 2005. The HYDRUS-1D software package for simulating the one-dimensional movement of water, heat, and multiple solutes in variably-saturated media. Department of Environmental Science, University of California, Riverside, CA.

Tartakovsky, D., 2013, Assessment and management of risk in subsurface hydrology: A review and perspective. Advances in Water Resources, 51: 247-260.

Tiedeman, C.R., M.C. Hill, F.A. D'Agnese, C.C. Faunt, 2003, Methods for using groundwater model predictions to guide hydrogeologic data collection, with application to the Death Valley regional groundwater flow system, Water Resour. Res. 39 (1), p. 5-1 to 5-17, doi:10.1029/2001WR001255

Tiedeman, C. R., D. M. Ely, M. C. Hill, and G. M. O'Brien, 2004, A method for evaluating the importance of system state observations to model predictions, with application to the Death Valley regional groundwater flow system, Water Resour. Res., 40, W12411, doi:10.1029/2004WR003313

Tiedeman, C. R., and C. T. Green, 2013, Effect of correlated observation error on parameters, predictions, and uncertainty, Water Resour. Res., 49, doi:10.1002/wrcr.20499.

Tonkin, M.J., Hill, M.C., and Doherty, John, 2003, MODFLOW-2000, the U.S. Geological Survey modular ground-water model – Documentation of MOD-PREDICT for predictions, prediction sensitivity analysis, and enhanced analysis if model fit: U.S. Geological Survey Open-File Report 03-385, 69 p. http://water.usgs.gov/nrp/gwsoftware/modflow2000/modflow2000.html

Tonkin, M.J., Tiedeman, C.R., Ely, D.M., and Hill, M.C., 2007, OPR-PPR, A computer program for assessing data importance to model predictions using linear statistics: U.S. Geological Survey Techniques and Methods, book 6, sec. E, chap. 2, 113 p. http://water.usgs.gov/software/OPR-PPR/

Tonkin, M.J., and J. Doherty. 2005. A Hybrid Regularized Inversion Methodology for Highly Parameterized Environmental Models: Water Resources Research. 41: W10412, doi:10.1029/2005WR003995.

Tonkin, M., and J. Doherty, 2009, Calibration-constrained Monte Carlo analysis of highly parameterized models using subspace techniques, Water Resources Res., 45, W00B10, doi:10.1029/2007WR006678.

Vrugt, J. A., C. J. F. ter Braak, M. P. Clark, J. M. Hyman, and B. A. Robinson (2008), Treatment of input uncertainty in hydrologic modeling: Doing hydrology backward with Markov chain Monte Carlo simulation, Water Resources Res., 44, W00B09, doi:10.1029/2007WR006720.

Vrugt, Jasper A. C.J.F. ter Braak, C.G.H. Diks, Bruce A. Robinson, James M. Hyman, Dave Higdon, 2009, Accelerating Markov Chain Monte Carlo Simulation by Differential Evolution with Self-Adaptive Randomized Subspace Sampling: International Journal of Nonlinear Sciences & Numerical Simulation,10(3),271-288.

Wagner B.J., and Gorelick, S.M., 1986. A statistical methodology for estimating transport parameters: theory and applications to one-dimensional advective-dispersive systems: Water Resources Research, v. 22, No. 8, p. 1303-1315.

Wainwright, H.M., Finsterle, S., Jung, Y., Zhou, Q., Birkholzer, J.T., 2014, Making sense of global sensitivity analyses. Computers & Geosciences 65 (84–94).

Wellman, J.F., Finsterle, S., Croucher, A., 2014, Integrating structural geological data into the inverse modeling framework of iTOUGH2. Computers and Geoscience, 65:95-109.

Welter, D.E., Doherty, J.E., Hunt, R.J., Muffels, C.T., Tonkin, M.J., and Schreüder, W.A., 2012, Approaches in highly parameterized inversion: PEST++, a Parameter ESTimation code optimized for large environmental models. U.S. Geological Survey Techniques and Methods, book 7, section C5, 47 p. http://wi.water.usgs.gov/models/pestplusplus/

Winston, R.B., 2000, Graphical User Interface for MODFLOW, Version 4: U.S. Geological Survey Open-File Report 00-315, 27 p. http://water.usgs.gov/nrp/gwsoftware/GW_Chart/GW_Chart.html

Yenihayat, Zehra, 1996, Damped least-squares inversion of confined aquifer pumping data based on singular value decomposition: Mathematical Geology, 28(2): 203-228

Appendix A: Connection with the Jupiter API and Other Open-Source Software

The JUPITER API [Joint Universal Parameter IdenTification and Evaluation of Reliability (JUPITER) Application Programming Interface (API)] is documented by Banta et al. (2006, 2008). New capabilities of UCODE_2014 are programmed using JUPITER API modules listed in Table A-1.

Table A- 1. JUPITER API modules, conventions, and mechanisms used to program the new capabilities in UCODE_2014.
[The modules listed are described in Banta et al. (2006).]

Module	Authorship
Datatypes	Banta
Global Data	Banta and Doherty
Utilities	Banta, Doherty, and Poeter
Basic	Banta and Doherty
Model Input-Output	Doherty and Banta
Dependents	Banta and Poeter
Equation	Doherty
Sensitivity	Banta and Poeter
Statistics	Poeter, Hill, and Banta
Prior-Information	Banta and Poeter
Parallel Processing	Banta
Conventions and Mechanisms	**Authorship**
Parameter-Value Generation	Hill, Banta, Poeter, and Doherty
Input and Output Specifications (including matrices)	Banta, Doherty, Hill, and Poeter
Input Block Names and Keywords	Doherty, Poeter, and Banta
Data-Exchange Files	Poeter, Banta, and Hill

MCMC was programmed in part by Dan Lu and in part using available public domain software, as shown in Table A-2.

Table A- 2. Subroutines used to implement MCMC in UCODE.

File dream.f90: All subroutines programmed by Dan Lu
Files rnglib.f90 and pdflib.f90: All subroutines are from http://people.sc.fsu.edu/~jburkardt/f_src/rnglib/rnglib.html with minor revisions by Dan Lu

Reference

Banta, E.R., Poeter, E.P., Doherty, J.E., Hill, M.C., 2006, JUPITER API: U.S. Geological Survey Techniques and Methods 6-E1, 268p. URL: http://water.usgs.gov/nrp/gwsoftware/jupiter/jupiter_api.html

Banta, E.R., **Hill, M.C**., E.P. Poeter, J.E. Doherty, and J. Babendreier, 2008, Building model analysis applications with the joint universal parameter identification and evaluation of reliability application programming interface (JUPITER API): Computers and Geoscience, 34: 310–319. doi:10.1016/j.cageo.2007.03.016

Appendix B: Files Produced by Using the Filename Prefix Specified on Command Lines from UCODE and Its Post-Processors

Several new files are produced using the fn filename prefix specified on the command line. For completeness and convenience, file extensions for all of the files are listed in Figure B-1 in alphabetical order with main output files listed first followed by data-exchange files.

Table B- 1. Files produced by UCODE_2014, RESIDUAL_ANALYSIS, RESIDUAL_ANALYSIS_ADV, LINEAR_UNCERTAINTY, MODEL_LINEARITY, MODEL_LINEARITY_ADV, and CORFAC_PLUS, and named using the fn prefix specified on the command line, in alphabetic order by letter in the file extension.

[File extensions that begin with # are main output files and are listed first. File extensions that begin with an underscore identify data-exchange files and are listed at the end of the table. Light gray shading identifies files produced by an auxiliary code to UCODE_2014; dark gray shading identifies files new with UCODE_2014. 'iteration' refers to parameter-estimation iteration. * is replaced by 'conf' for results related to confidence intervals or 'pred' for results related to prediction intervals.]

Extension	Brief description (see also Tables 16-22, 24-28, 34, 35 of UCODE_2005 Manual)	Content[1]	Use[1]
The following are main output files for the listed auxiliary codes			
#corfac_*	CORFAC_PLUS	35	34
#linunc	LINEAR_UNCERTAINTY	26	--
#modlin	MODEL_LINEARITY	27	--
#modlinadv	MODEL_LINEARITY_ADV	--	--
#resan	RESIDUAL_ANALYSIS	24	--
#resanadv	RESIDUAL_ANALYSIS_ADV	--	--
The following are UCODE_2014 main output files for the listed modes			
#ucreateinitfiles	Sensitivity-analysis with CreateInitFiles=yes	--	--
#umcmc	MCMC simulation for parameter samples	--	--
#umcmc_pred	MCMC simulation for prediction samples	--	--
#umodlin	Test-model-linearity	--	--
#unonlinint_*	Nonlinear-uncertainty	--	--
#uout	Forward, sensitivity-analysis, parameter-estimation	--	28-30
#upred	Prediction	20	33
#usos	Evaluate-objective-function	--	--
The following are data exchange files			
_b1	Parameter sets for _b2	19	--
_b1adv*	Parameter sets for _b2adv*	35	34
_b2	Values simulated using parameter sets from _b1	19	--
_b2adv*	Values simulated using parameter sets from _b1adv*	35	34
_b3*	Parameter sets for _b4*	35	34
_b4*	Values simulated using parameter sets from _b3*	35	34

Appendix B: Files Produced by Using the Filename Prefix Specified on Command Lines from UCODE and Its Post-Processors

Extension	Brief description (see also Tables 16-22, 24-28, 34, 35 of UCODE_2005 Manual)	Content[1]	Use[1]
_cf*	Correction factors	35	34
_cfsu	Sensitivities used by CORFAC_PLUS	35	34
_dm	Information about model structure, fit and parsimony	21	--
_dmp	Information about predictions	21	--
_dm_presvd[6]	Copy of _dm based on parameters used to start SVD	21	ra
_gm	Observation groups	16	--
_gmp	Prediction groups	20	--
_init	As for _paopt evaluated at other parameter values	15	--
_init._mv	As for the _mv file evaluated at other parameter values	15	--
_init._su	As for the _su file evaluated at other parameter values	15	--
_init._supri	As for the _supri file evaluated at other parameter values	15	--
_int*	Nonlinear intervals	35	34
_int*par	Parameter values at nonlinear interval limits	35	34
_int*wr	Weighted residuals at nonlinear interval limits		
_linp	Predictions and their linear confidence intervals	26	--
#linunc	LINEAR_UNCERTAINTY main output file	26	--
_mc	Parameter correlation coefficient matrix	19	30
#modlin	MODEL_LINEARITY main output file	27	--
#modlinadv	MODEL_LINEARITY_ADV main output file		
_mv	Parameter variance-covariance matrix	19	30
_mv_presvd[6]	Copy of _mv based on parameters used to start SVD	19	ra
_mcmc_par##	Parameter samples of chain##	--	--
_mcmc_sim##	Simulated values of chain##		
_mcmc_grr	Gelman-Rubin R statistics	--	--
_mcmc_pred##	Prediction samples of chain##	--	--
_mcmc_restart	Information to restart the MCMC run	--	--
_nm	Weighted residuals, probability plotting positions	16	31
_os	Unweighted simulated and observed or prior values	16	31
_p	Predictions	20	--
_pa	Parameter values for each iteration	19	--
_paopt [6]	Information for all defined parameters	19	--
_paopt_presvd [6]	Copy of _paopt based on parameters used to start SVD	19	svd/ra
_pasub	Parameter values for each iteration	19	--
_pc [6]	Information for estimated parameters	19	30

Appendix B: Files Produced by Using the Filename Prefix Specified on Command Lines from UCODE and Its Post-Processors

Extension	Brief description (see also Tables 16-22, 24-28, 34, 35 of UCODE_2005 Manual)	Content[1]	Use[1]
_pcc	Large parameter correlation coefficients (≥ 0.85)	18	30, 31
_pr	Prior information equations	16	--
_pv	Prediction variances	20	--
_r	Unweighted residuals (observations and prior)	16	31
_rb	DFBetas statistics for each observation and parameter	24	32
_rb_presvd	DFBetas statistics for results used to start SVD	24	32
_rc	Cook's D statistic for each observation	24	32
_rc_presvd	Cook's D statistic for results used to start SVD	24	32
_rd	Ordered uncorrelated random numbers	24	32
_rd_presvd	Ordered uncorrelated random numbers for results used to start SVD	24	32
_rdadv	Results of RESIDUAL_ANALYSIS_ADV	--	--
_rg	Ordered correlated random numbers	24	32
_rg_presvd	Ordered correlated random numbers for results used to start SVD	24	32
_s1	One-percent scaled sensitivities	18	30, 31
_sc	Composite scaled sensitivity	18	30, 31
_sc_svd[6]	Decomposition of CSS using SVD of scaled [wt^0.5][X]	--	--
_sc_svd_presvd[6]	Copy of _sc_svd based on parameters used to start SVD	--	--
_sd	Dimensionless scaled sensitivities	18	30, 31
_so	Sensitivity summary by observation, including leverage	18	30, 31
_sos	Parameter values and resulting value of the sum of squared weighted residuals objective function.	19	--
_sppp	Prediction sensitivity scaled by Param[5]/PredValue[4]	20	--
_sppr	Prediction sensitivity scaled by Param[5]/RefValue[3]	20	--
_spsp	Prediction sensitivity scaled by PSD[2]/PredValue[4]	20	--
_spsr	Prediction sensitivity scaled by PSD[2]/RefValue[3]	20	--
_spu	Unscaled sensitivities for predictions	20	--
_ss	Sum of squared weighted residuals by iteration	19	31
_su	Unscaled sensitivities for observations	18	--
_su_presvd[6]	Copy of _su based on parameters used to start SVD	18	ra
_summary	Summary of regression progress	--	--
_summary_presvd	Summary of regression progress for run to start SVD	--	--
_supri	Unscaled sensitivities for prior information	18	--
_supri_presvd	Copy of _supri based on parameters used to start SVD	18	ra
_svd	Singularvalues and vectors of [wt^0.5][X]{b} matrix	--	--
_svd_presvd	Copy of _svd based on parameters used to start SVD	--	--
_svd-id	SVD-ID statistics for a range of SVDnumber	--	--
_svd-id_presvd	Copy of _svd-id based on parameters used to start SVD	--	--

Appendix B: Files Produced by Using the Filename Prefix Specified on Command Lines from UCODE and Its Post-Processors

Extension	Brief description (see also Tables 16-22, 24-28, 34, 35 of UCODE_2005 Manual)	Content[1]	Use[1]
_w	Weighted residuals, observations and prior information	16	31
_w_presd	Copy of _w based on parameters used to start SVD	16	ra
_ws	Simulated equivalents and weighted residuals for observations and prior information	16	31
_wt	Weights for observations	17	--
_wt_presvd	Copy of _wt based on parameters used to start SVD	17	ra
_wtpri	Weights for prior information	17	--
_wtpri_presvd	Copy of _wtpri based on parameters used to start SVD	17	ra
_ww	Weighted simulated equivalents in relation to weighted observations or prior information	16	31

[1] Tables in the UCODE_2005 report that describe the file contents and suggest how to use it. A --, indicates the content or use of the file is not described in a table. Svd indicates the file is needed to start a subsequent SVD parameter estimation. Ra indicates the file is needed to execute Residual_Analysis on the residuals of the run used to start the SVD.

[2] Parameter Standard Deviation.

[3] Reference Value from Prediction_Data block. Scaled sensitivity is set to zero if this number equals zero.

[4] Predicted Value. Scaled sensitivity is set to zero if this number equals zero.

[5] Parameter Value.

[6] These data-exchange files are not generated from JUPITER API subroutines; they are part of UCODE_2014.

Appendix C: Distributed Directories

UCODE_2014 and the post-processors can be downloaded from the web site listed in the preface. The operating system is listed for each compiled downloadable executable file. When uncompressed, a directory is created with four subdirectories. The directory is named wrdapp\ucode_2014. The subdirectories are listed in Table C-1.

Table C- 1. Contents of the directories distributed with UCODE-2014.

Subdi-rectory	Contents
bin	Executable file.
doc	This documentation and the UCODE_2005 documentation, in PDF format.
src	Fortran source files organized into 11 directories. All source files are named with the extension “f90”. Except for directory API-MODULES, each directory contains a file named readme.txt that lists the modules needed to compile the code to which the directory is dedicated. The subdirectories are: API-MODULES: Jupiter API modules used in the codes listed below. UCODE_2014: Files unique to UCODE_2014. JRUNNER: Files unique to the computer code JRUNNER, which needs to be run on machines to be used for parallel computations. Eight directories each named for one of the other codes documented in this report or the UCODE_2005 report. Each of the directories contains the files unique to the code involved.
test-*win*	Files for running examples for the SVD and MCMC capabilities documented in this report and for running the models in Chapter 9 in a windows operating system. The contents of this directory are discussed in Appendix D.

Table C- 2. Files distributed with UCODE_2014 relevant to MCMC.
[fn is replaced by HO1 in the test case I and HO2 in the test case II, as defined on the execution lines of UCODE_2014 and the MCMC program.]

FILENAME	DESCRIPTION
Executable Files – in directory bin	
UCODE_2014.exe	Run UCODE_2014 MCMC simulation
mf2005.exe	Run forward model
MCMC Modules – in directory src/ucode_2014	
rnglib.f90	Generate random numbers from U[0,1] uniform distribution
pdflib.f90	Generate random numbers from a certain distribution and calculate density values for that distribution
dream.f90	Read in input, run DREAM algorithm, print out results.

Appendix D: Test Cases Distributed with UCODE_2014

Test cases for the SVD and MCMC capabilities described in this report and files for the model analyses in Chapter 9 are distributed with UCODE_2014 in subdirectories of the test-win directory (Table C-1). The contents of this directory are listed in Table D-1.

SVD Test Case and Selected Results

The SVD test case is based on the test case used for exercises in Hill and Tiedeman (2007), modified so that the top boundary is a free surface water table and in the "true" synthetic system the top layer is assigned a randomly generated heterogeneous hydraulic conductivity. The problem is described in Chapter 9 of this report. The conceptual model, numerical model, observations, and simulation of model features are shown in Figures 18 to 20. The true hydraulic conductivity field is shown in Figure 23. The distributed directory structure and the batch files for the SVD test cases are listed in Table D-1. This section presents selected results from the SVD test case and discusses them briefly.

The test case presented here has fewer parameters than observations. Often SVD is used to solve highly parameterized inverse problems, roughly defined as having more parameters than observations. UCODE_2014 does not support simulation of highly parameterized models and users are encouraged to use PEST for this purpose. In general, it is often useful to simulate a model constructed using few parameters and conduct sensitivity analyses and preliminary parameter-estimation runs before proceeding to highly parameterized models. UCODE_2014 provides extensive sensitivity-analysis statistics to be used in such investigations. Identified characteristics such as insensitive parameters and highly correlated parameters can be used to manage the highly parameterized simulation. For example, such an analysis might be used to justify highly parameterizing sensitive aspects of the system and not estimating any parameters associated with aspects of the system to which observations provide no information.

Directory SVD-Test (Table D-1) includes the model B calibration run described in Chapter 9. The results presented in Chapter 9 are from a traditional UCODE regression approach in which the values of the process-model parameters are estimated directly by regression. Directory SVD-Test also includes two runs for which SVD is used in the estimation. In a run with SVDupdate=SVDall, SVD null-space parameters are set to create a tractable regression. In a run with SVDupdate=Option2_SVDset or Option2_css_pcc, the same number of process-model parameter values are set to create a tractable regression problem, with the parameters selected using PCC and CSS (see Chapter 4 for more information). Setting process-model parameter values clarifies the consequences of reducing the parameter space – modelers can easily see that some parameter values are maintaining their original value and can check to see which of these parameters has large PCC or small CSS. If the choices and PCC and CSS values do not make sense based on the understanding of the system, modelers can then consider potential mistakes in the UCODE input file and reconsider things like model construction, parameterization, and observation representation in the model.

Table D- 1. Contents of directory Test-win. SA, sensitivity analysis. Input files for conducting runs are provided; directories also will contain output files after the batch files are run.

Name of directory	
Chap4-test-SVD	
Batch files, MODFLOW files, UCODE_2014 input files, and auxiliary files for forward, SA, regression (with and without use of SVD methods), and residual analysis runs of model B in Chapter 9. The SVD regression runs use two options: Option2_css_pcc and SVDall. The residual analysis run is conducted for the PMP (process-model parameters) regression without use of SVD, because the SVD model runs produce very large values of parameter standard deviations (Table D-2) and cannot be processed using Residual_Analysis. This is common when SVD truncation is used to obtain a tractable regression problem. See the file *_00-READ-ME-SVD-test.txt* for a description of all runs.	
Chap5-test-MCMC	
Files for MCMC test cases. See the file *_00-READ-ME-Chap5-test-MCMC.txt*.	
Subdirectory	Contents
HO1/HOmodel_test1 (Runner directories normally removed for distribution. Create using the provided batch file.)	Batch files, UCODE_2014 input files, and auxiliary files. Folders related to forward and prediction run. After the programs are run it includes two runner directories (Runner1 and Runner2) created in the HO1 subdirectory. This test case provides an example of how to set up a parallel run.
HO2/HOmodel_test2	Batch files, UCODE_2014 main input files, and auxiliary files. Folders related to forward and prediction run.
Notes: For HO1 and HO2, the MCMC batch files are in ***HOmodel_test****, where ***** is ***1*** or ***2***. The batch files are numbered from 01 to 05; run them sequentially.	
Chap9-Faux Valley	
Files for synthetic Faux Valley groundwater flow model SA, regression, and analysis. For a complete description of the runs, see the file *_00-READ-ME-chap9-FauxValley.txt*, and the *_00-READ-ME-*.txt* file within each subdirectory, where * is the subdirectory name. These runs use only UCODE_2005 capabilities of UCODE_2014.	
Subdirectory	Batch files, UCODE_2014 input files, and auxiliary files for:
01-ModelA-InitialSensAnalysis	SA run for model A with initial parameter values
02-ModelB-Prediction	SA, prediction, and prediction uncertainty runs for calibrated model B (regression run for Model B is in ***Chap4-test-SVD***)
03-ModelA-SensAnalysis-ParameterSet2	SA run for model A with parameter values set to lower bounds of reasonable ranges.
04-ModelB-SOS	Sum-of-squares surface run for model B.
05-ModelB-SensAnalysis	SA run for calibrated model B with all parameters active and no prior.
06-ModelB-Prediction-SOS	Sum-of-squares surface run for model B with predictions.
07-ModelB-opr-add	OPR-PPR runs for model B, with mode OPRADD.
08-ModelB-ppr	OPR-PPR run for model B, with mode PPR.
truth	MODFLOW runs with input files representing the true system

Name of directory	
convert_pest_prior_info	
Batch file that runs convert_pest_prior_info which reads the pest input file provided in the directory (pestrun-p.pst), finds the prior information items in that file and creates a UCODE Linear_Prior_Information input block which is written in the file pestrun-p.pst.out. The output file ConvertPestPriorInfo.#out summarizes what was done. The _clean,bat file removes the output.	
Exercises from Hill and Tiedeman 2007	
Batch files, MODFLOW files, UCODE input files, and auxiliary files for runs that use only UCODE_2005 capabilities of UCODE_2014. See file *_00-READ-ME-ExercisesHillandTiedeman.txt*.	
simple-example	
Batch and input files to run the dupuit model parametyer estimation. The problem is described in the A-description-of-the-dupuit-example.htm file that is included in the folder which uses files in the directory html. The dupuit code source and executable reside in the code_dupuit directory.	
ucode_2005_examples_run_with_ucode_2014	
Files for running UCODE_2014 for two of the examples distributed with UCODE_2005. See file *readme.txt*.	
Subdirectory	Contents
_test-win-2005	UCODE batch files (in directories ***ex1a*** and ***ex1b***) and input files (in directories ***ex1a-files*** and ***ex1b-files***).
test-data-win	MODFLOW batch files and input files (in directories ***data-****).

Use of SVD for this relatively simple problem provides the opportunity to investigate SVD performance at a level of detail that is difficult when highly parameterized models, for which the number of parameters exceeds the number of observations, are considered. Here, we investigate SVDupdate.

Table D-2 shows selected results from the three UCODE_2014 runs distributed in the SVD-test directory. For SVDupdate=SVDall, null-space SVD parameters are held constant and all the process-model parameter (PMP) values change. For SVDupdate=Option2_SVDset or Option2_css_pcc, the values of selected process-model parameters do not change.

For the PMP model run, the user must decide which process-model parameter values will be set and which parameters will include prior information, to attain a tractable regression problem. These decisions usually result from inspection of sensitivity analysis (SA) results, model fit, and preliminary regression runs. This process is ad hoc and SVD provides a more formal mechanism for achieving a tractable regression problem.

When using SVD with SVDupdate=SVDall, Option2_SVDset, or Option2_css_pcc, generally no preliminary analysis is needed for a given model and the best-fit parameters listed in Table D-2 were obtained with one run of UCODE_2014 with SVD=yes. The best-fit parameter values are of the same magnitude for all three parameter-estimation runs, with the exception of HK_1_2, which has an unrealistic value of 48 for the SVD run with SVDupdate=SVDall. This illustrates the importance of carefully evaluating the parameter estimates following any UCODE run.

The bottom half of Table D-2 shows that the standard deviations of the parameters estimated with SVD are very large. This is typical of SVD runs and occurs because UCODE (and other codes using

SVD) depends on the SVD process to set parameter values and calculates the final statistics with all parameters active. The only information provided is from the observations. The resulting standard deviations would be considered reasonable only if there were no additional independent information on any of the parameter values, which is unlikely. Thus, the standard deviations calculated for the SVD model runs are too large, indicating a greater level of uncertainty than is supportable by data and knowledge.

In the PMP model runs some parameters are set and some parameters have prior information included. The setting of parameters is equivalent to saying that their values are known completely, which is not realistic. Thus, the standard deviations calculated for the PMP model are too small, indicating a greater level of certainty than is supportable by data and knowledge.

How can realistic evaluation of uncertainty be obtained? Hill and Tiedeman (2007) suggest activating all defined parameters and including any available knowledge about each parameter as uncertain prior information. The result of such a model run is shown in Table D-2. While clearly more reasonable, uncertainty in the parameterization is not accounted for in these results, which would suggest the uncertainty measures continue to be somewhat small. This could be evaluated using multiple alternative realizations of the parameterization using software such as MMA (Poeter and Hill, 2007), models that are highly parameterized using pilot points with software such as PEST or PEST++ (Doherty, 2010; Welter et al., 2012), or Bayesian methods such as bgaPEST (Fienen et al., 2013).

Table D-3 shows graphs displaying the model fit, sensitivity analysis, and changes in parameter values during regression for three calibration approaches. Of note are the following.

- The most and least important parameters are consistent, while the intermediate parameters vary (from analysis of CSS)
- Correlation between hydraulic conductivity and recharge parameters is common (from analysis of PCC)
- The flow is an important observation (from analysis of leverage). Heads at locations 9 and 7 in layer 2 and 1 in layer 1 tend to be important.

Table D- 2. (A) True and estimated parameters and (B) observed and simulated heads and flows for model runs in the SVD-Test directory.

[PMP, process-model parameter. SVD, singular value decomposition. SOSWR, sum of squared weighted residuals. SVDupdate is described in Chapter 4 of this report. The parameter names (HK_1_1, HK_1_2, and so on), are described in Chapter 9. Infinity, the exponent was so large it could not be evaluated. Shading: Parameter value is not allowed to change. Prior information is defined for this parameter.]

(A)				SVD	
				SVDupdate	
	TRUE	[1]PMP	[2]PMP	[3]SVDall	[3]Option2 _css_pcc
Parameter values					
HK_1_1 (m/s)	Variable	2.6E-04	Same values as in column to the left	2.6E-04	2.7E-04
HK_1_2 (m/s)	Variable	1.0E-02		48	5.4E-02
HK_2 (m/s)	4.40E-05	3.2E-05		1.8E-05	3.3E-05
VANI (--)	1	1		1.0	1
VK_CB (m/s)	2.00E-07	1.00E-07		4.8E-08	1.00E-07
K_RB (m/s)	1.00E-03	1.20E-03		8.1E-04	1.20E-03
RCH_1 (cm/yr)	40	50		46	47
RCH_2 (cm/yr)	70	80		79	87
[4]Parameter standard deviation calculated as $s^2(\mathbf{X}^T\omega\mathbf{X})^{-1}$					
HK_1_1 (m/s)		2.7E-04	1.4E-04	9.2E-04	1.1E-03
HK_1_2 (m/s)		65	1.6	Infinity	Infinity
HK_2 (m/s)		5.2E-05	1.4E-02	4.2E+92	6.3E+40
VANI (--)		--	0.6	Infinity	Infinity
VK_CB (m/s)		--	9.9E-05	1.4E+96	8.1E+40
K_RB (m/s)		--	2.5E+06	8.2E+27	1.5E+59
RCH_1 (cm/yr)		50	36	2.3E+04	3.9E+06
RCH_2 (cm/yr)		50	38	1.3E+03	4.7E+03
Standard error of the regression (s in the equation used to calculate the parameter standard deviation)					
Value		7.6	--	11	12
[5]Confidence interval		5; 15	--	6; 42	7;43

[1] Model B of Chapter 9.

[2] Model PMP results with all parameters active, prior information applied, and standard deviations calculated at the estimated values (using a Sensitivity Analysis run of UCODE_2014).

[3] For these runs all parameters are log-transformed.

[4] From the _pc file column labeled "STANDARD DEVIATION (NATIVE)". The large (and infinite) values occur because no prior is defined for the SVD runs and no parameters are set for the calculation of parameter uncertainty. Because of the lack of prior, these values calculated for SVD model runs are not very useful. Parameter prior information such as that shown in Table 5 needs to be included in the calculations to obtain meaningful standard deviations.

[5] If the value 1.0 is not included in the confidence interval, this indicates there is model error. See Hill and Tiedeman (2007, p. 96-98).

Table D-2-continued. (A) True and estimated parameters. (B) SOSWR and observed and simulated heads and flows. For model runs in the SVD-Test directory.

[Shaded numbers: Head residuals with absolute values greater than 1.0 m, highlighted to illustrate the largest residuals.]

(B)				SVD			
				SVDupdate			
		PMP		SVDall		Option2_SVDset	
SOSWR (obs)		401		422		374	
SOSWR(prior)		5		--		--	
Observation name	Observed	Simulated	Residual	Simulated	Residual	Simulated	Residual
hd01.ss	100.25 m	100.31	-0.06	100.44	-0.19	100.32	-0.07
hd02.ss	149.41 m	148.70	0.71	148.87	0.54	148.79	0.62
hd03.ss	163.46 m	164.43	-0.96	163.94	-0.48	164.11	-0.65
hd04.ss	153.75 m	148.70	5.05	148.87	4.88	148.79	4.96
hd05.ss	160.36 m	163.77	-3.41	163.94	-3.58	163.98	-3.62
hd06.ss	148.90 m	148.53	0.37	148.64	-0.26	148.62	0.28
hd07.ss	103.77 m	103.29	0.48	103.86	-0.09	103.38	0.39
hd08.ss	163.87 m	164.55	-0.68	164.12	-0.25	164.26	-0.38
hd09.ss	180.67 m	179.33	1.34	179.55	-1.12	179.41	1.26
hd10.ss	160.80 m	162.28	-1.48	162.19	-1.39	162.38	-1.58
Flow01_1	-5.77 m^3/s	-6.72	0.95	-6.43	0.66	-6.87	1.10

Table D- 3. Graphs showing results of model fit and sensitivity analysis for the starting parameters and three calibrated models, and parameter value changes for the calibrated models.

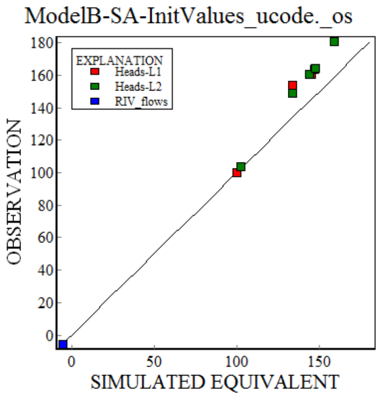
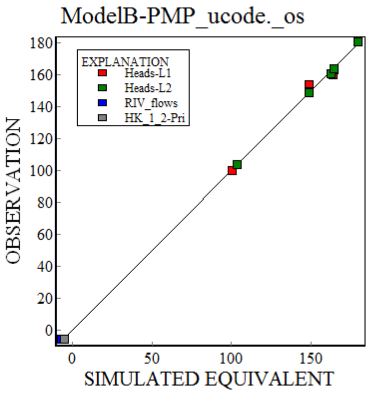
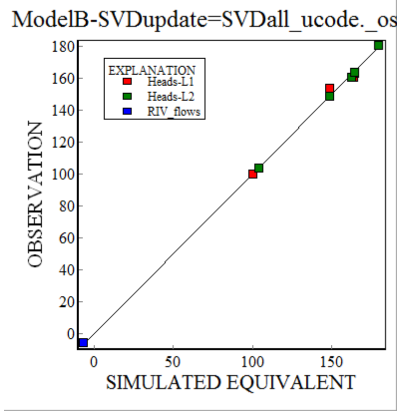
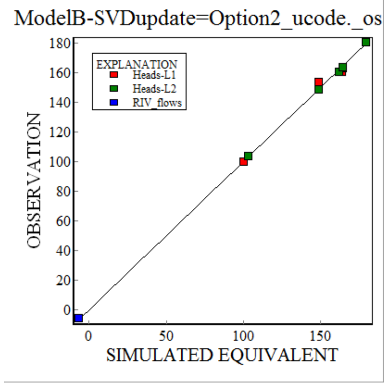
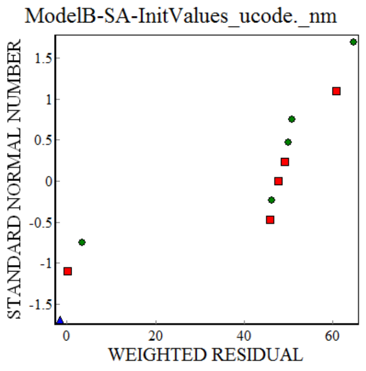
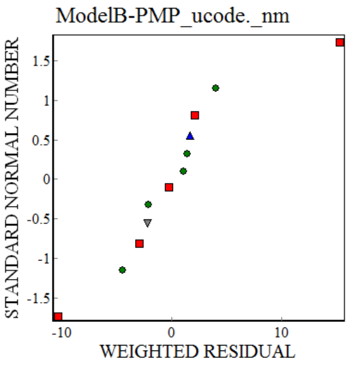
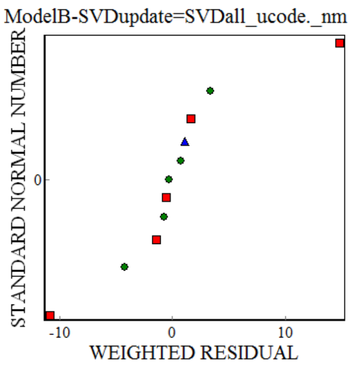
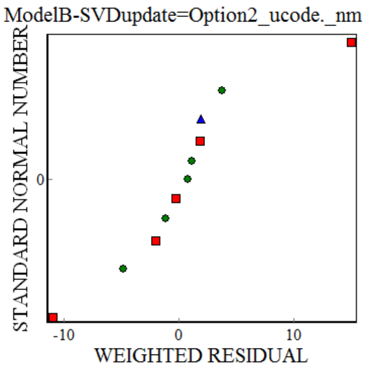
Table D-3	Calibrated (PMP, process-model parameters; SVD singular value decomposition)		
		SVDupdate	
Starting parameters	PMP	SVDall	Option2_SVDset
Model Fit			
			
			
Points at upper right and bottom left for calibrated models (PMP and SVD) are hd04 and hd05, respectfully. Locations are shown in Figure 19. Problem for the reader: Why might the model have a hard time fitting these two observations? (Hint: distance from stream)			

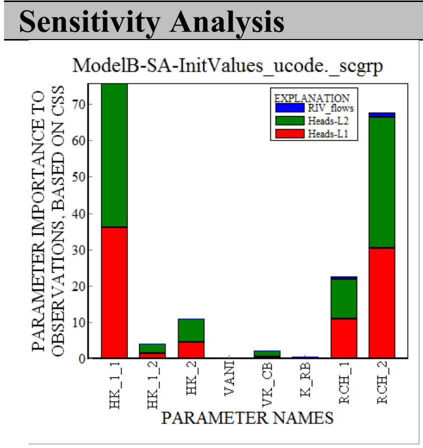
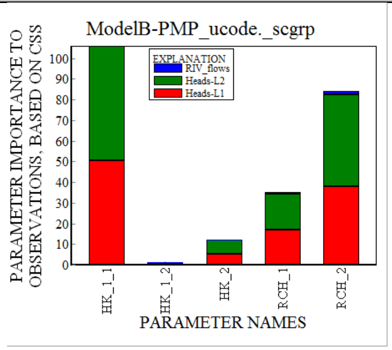
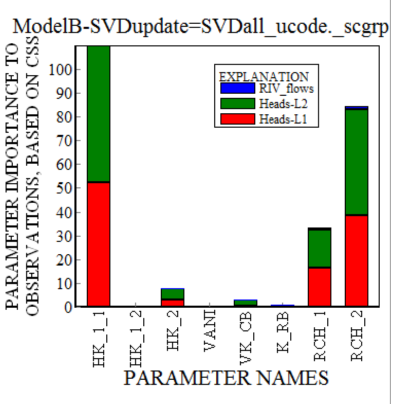
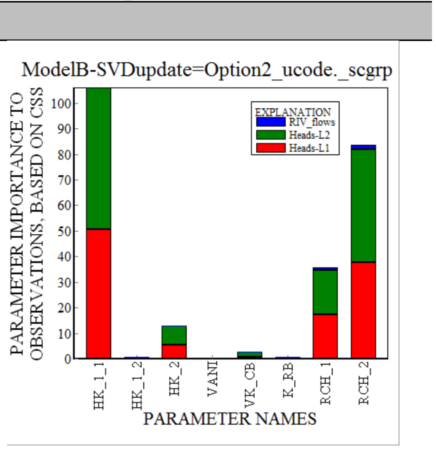
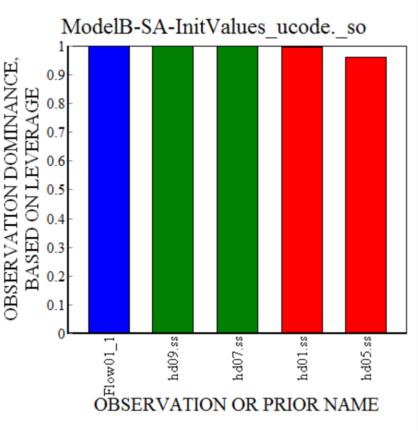
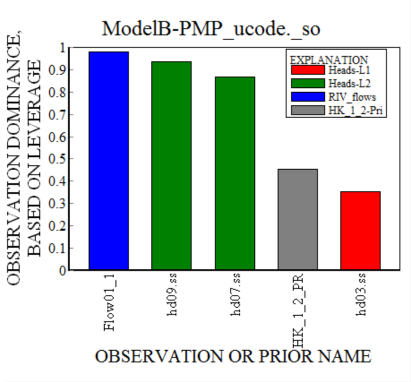
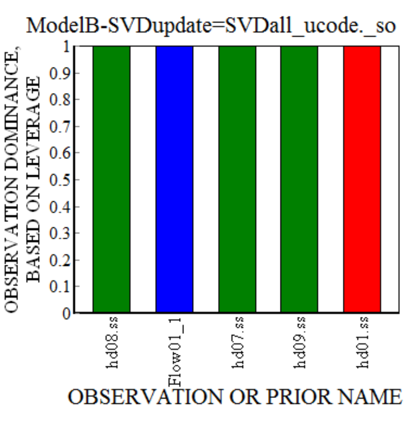
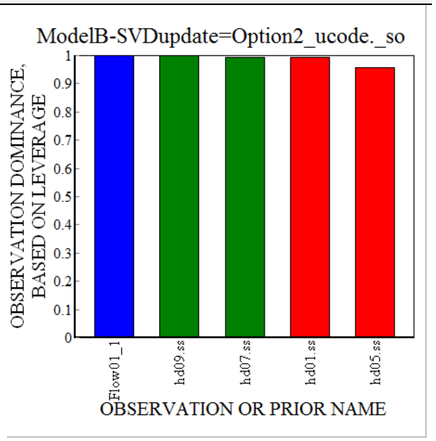
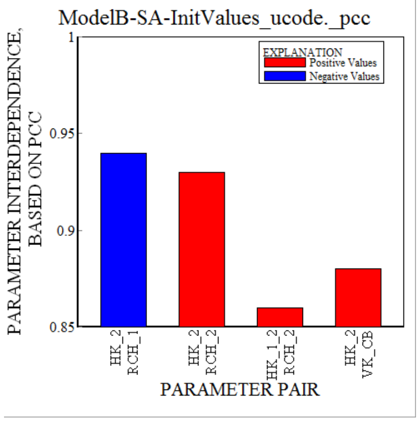
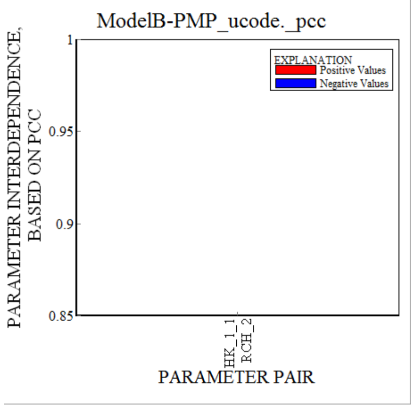
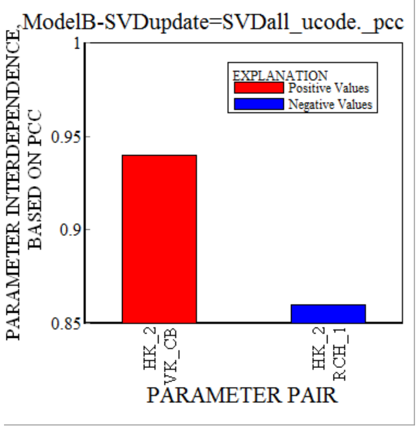
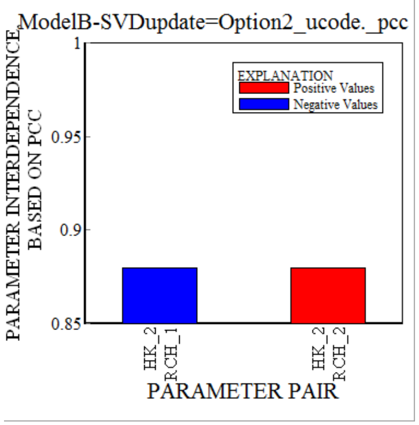
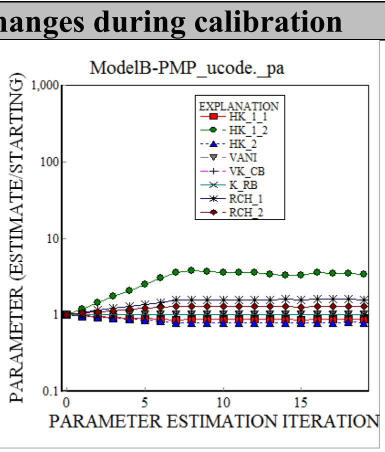
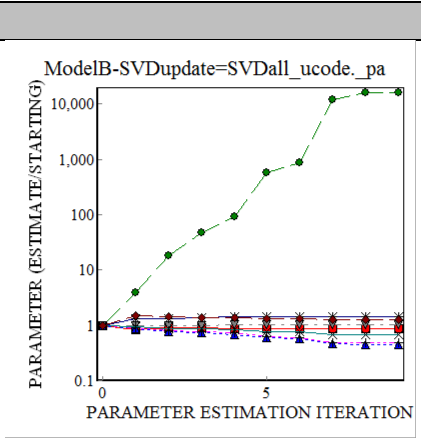
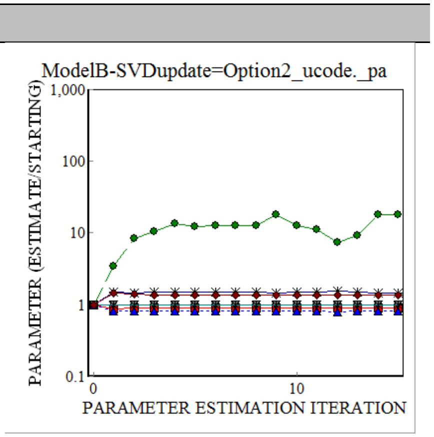
Table D-3	Calibrated (PMP, process-model parameters; SVD singular value decomposition)		
		SVDupdate	
Starting parameters	PMP	SVDall	Option2_SVDset
Sensitivity Analysis			
	 *Set parameters not included.*		
For all cases, HK_1_1 and RCH_2 are most important.			
	 *hd01 and hd05 not important given the set parameters and prior on HK_1_2.*		
Leverage=1 means the one observation potentially controls the value of one or more parameter. Largest five values shown.			

Table D-3	Calibrated (PMP, process-model parameters; SVD singular value decomposition)		
		SVDupdate	
Starting parameters	PMP	SVDall	Option2_SVDset
Sensitivity Analysis			
			
PCC often varies. Look for parameter pairs with PCC that are consistently greater than 0.95.			
Process-model parameter value changes during calibration			
No panel included because no estimation was conducted			
The parameter values are scaled by the starting parameter value so changes are relative to the starting values.			

MCMC Example Simulations

The MCMC test cases are derived from simulations used in Lu et al. (2012), which were derived from Hill et al. (1998). The true synthetic system is shown in Figure D-1 and includes an areally variable hydraulic conductivity field. The calibrated model uses a homogeneous hydraulic-conductivity field throughout the system except where the confining unit and lake are shown in Figure D-1(a). The model has three layers.

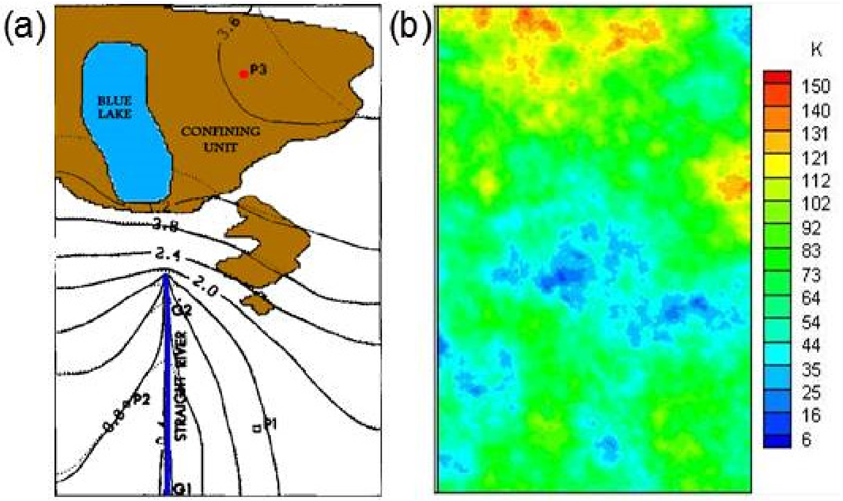


Figure D- 1. (a) Modeling domain with the confining unit, Blue Lake, and Straight River, (b) the true horizontal hydraulic conductivity field (K field) with values from 6 to 150 m/d.

There are two examples, called HO1 and HO2. For HO1, each parameter has a uniform prior probability distribution. Also, the model is set up to run in parallel. For HO2, five parameters have a multinormal prior distribution and one is uniform. Both examples are set up to conduct an MCMC run and also to calculate predictions.

The parallel setup for HO1 is likely to be useful for other applications. Batch files are used to create the runner directories and start the runners. These are files 01-runners-populate.bat and 02-start-runners.bat, and need to be run in that order. For distribution the runner directories are removed.

To run the MCMC code in Windows system, do one of the following:

- Double click the batch file names. Note that for these two test problems, the maximum samples for each chain is set to 20, so users can get the simulation results after a minute.
- In a command window, type the contents of the batch file to run the simulation.

Figures D-2 through D-9 show selected results from HO1, including files from the distributed example and results from a much longer run.

```
"MONITORED PARAMETER SAMPLE VALUES AND ASSOCIATED LOG LIKELIHOOD FUNCTION VALUES FOR CHAIN # 01"
"Iteration"   "LogLikelihood"    "     LAKERCH"  "         KRB0"  "          RCH"  "           KV"
      1      -6.1284370E+01       4.8014610E-04   5.4413374E+03   8.4593439E-04   2.9612086E-03
      2      -6.1284370E+01       4.8014610E-04   5.4413374E+03   8.4593439E-04   2.9612086E-03
      3      -6.1284370E+01       4.8014610E-04   5.4413374E+03   8.4593439E-04   2.9612086E-03
      4      -6.1284370E+01       4.8014610E-04   5.4413374E+03   8.4593439E-04   2.9612086E-03
      5      -6.1284370E+01       4.8014610E-04   5.4413374E+03   8.4593439E-04   2.9612086E-03
   .
   .
   .
     18      -5.9724230E+01       1.0670284E-03   5.1639845E+03   7.8059344E-04   8.0438356E-01
     19      -5.9724230E+01       1.0670284E-03   5.1639845E+03   7.8059344E-04   8.0438356E-01
     20      -5.9724230E+01       1.0670284E-03   5.1639845E+03   7.8059344E-04   8.0438356E-01
```

Figure D- 2. First 6 columns of the MCMC output file HO1._mcmc01 for chain 1. For any actual application this file would include thousands and perhaps 10,000s of lines.

```
   "PREDICTION VALUES CALCULATED FROM PARAMETER SAMPLES AFTER
   CONVERGENCE FOR CHAIN # 01"
    "Number"    "   G2_change"  "          DD"
          1       3.5876111E-01   1.3824296E+00
          2       3.5876111E-01   1.3824296E+00
          3       3.5876111E-01   1.3824296E+00
          4       3.5876111E-01   1.3824296E+00
          5       3.5876111E-01   1.3824296E+00
          6       3.5876111E-01   1.3824296E+00
          7       3.6098343E-01   1.2867179E+00
          8       3.7380822E-01   1.4330068E+00
          9       3.7380822E-01   1.4330068E+00
         10       3.7380822E-01   1.4330068E+00
```

Figure D- 3. Output file HO1._mcmc_pred01 from MCMC test case 1 for the example 10 parameter sets following the 10 sets ignored based on user input. For any actual application this file would include thousands and perhaps 10,000s of lines.

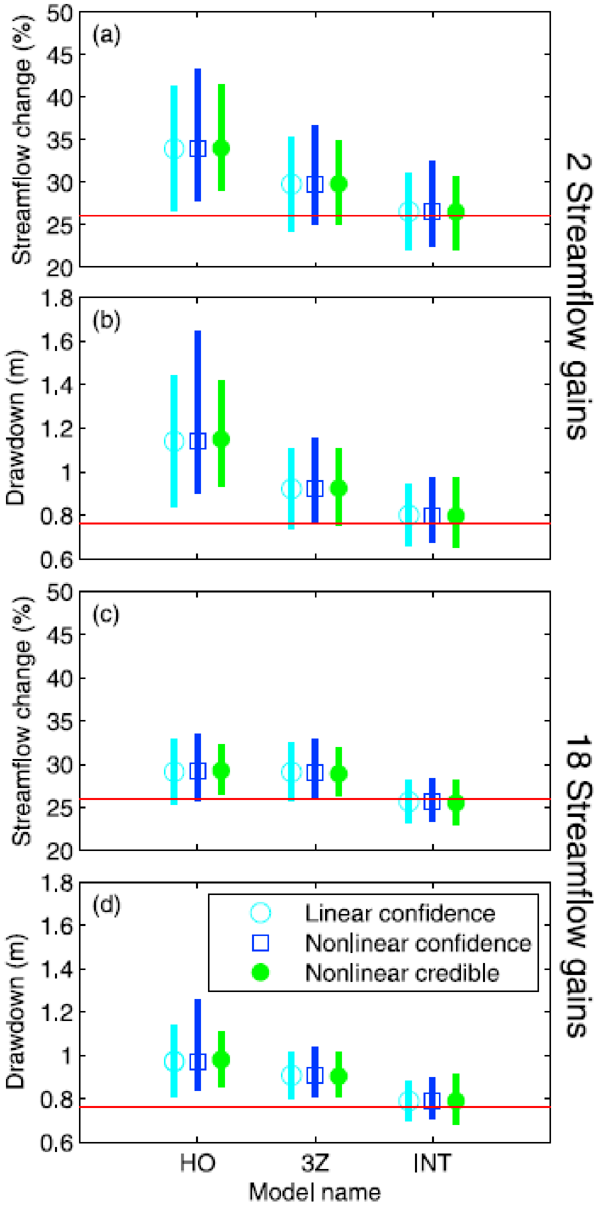


Figure D- 4. Figure showing MCMC credible intervals plotted from files such as HO1._mcmc_pred01. Also shown are linear and nonlinear confidence intervals which can also be calculated using UCODE_2014, for the MCMC test case HO1. The credible intervals required 420,000 models runs. The linear and nonlinear confidence intervals required ≤106 and ≤1494 model runs, respectively. (from Lu et al., 2012)

Analysis of MCMC results with GW-Chart

GW_Chart supports the UCODE MCMC capability as demonstrated in this section. The plots in Figures D-5 through D-9 were produced by saving plots from GW_Chart.

Based on output files HO1._mcmc_par** (where ** represents chain index, i.e., 01, 02, … 06) the evolution of samples from all chains can be plotted using software GW-Chart like Figure D-5. This plot can reveal four characteristics: (1) the chains' evolutions to the target posterior distribution, (2) a rough estimate of whether and when the chains are mixed, (3) whether the parameter posterior distribution has multiple modes, and (4) how the chains jump between the modes.

The parameter samples saved in HO1._mcmc_par** files can also be used to construct histogram using GW-Chart like Figure D-6. The histogram constructed with the converged parameter samples can estimate the marginal posterior distribution of the parameter.

The plot of the Gelman-Rubin R statistics saved in file HO1._mcmc_grr can be used to diagnose chain convergence. The plot made in GW-Chart is shown in Figure D-7.

The prediction values simulated by converged parameter samples are saved in HO1._mcmc_pred** files. Based on these values, the histogram plotted with GW-Chart in Figure D-8 can be used to estimate the posterior distribution of the prediction. The probability plot of GW-Chart in Figure D-9 evaluates the Bayesian credible intervals. Figure D-9 shows two of the credible intervals shown in Figure D-4 a and b.

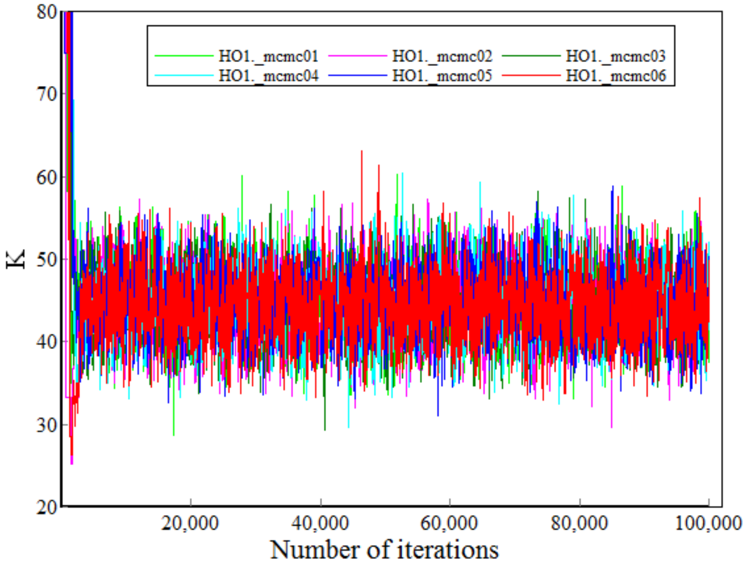


Figure D- 5. Evolution of sampled K values in six chains.

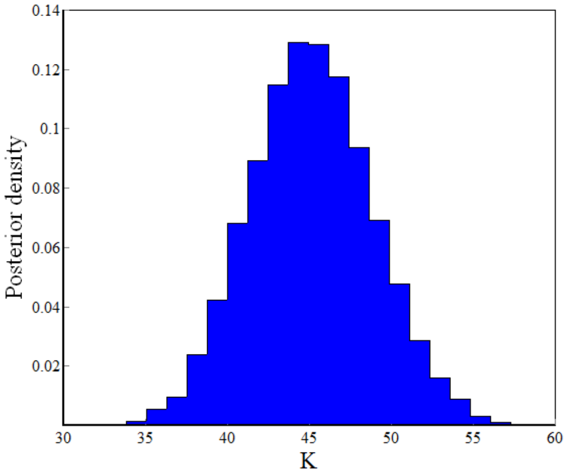


Figure D- 6. Simulated posterior density of K based on samples after convergence (the last 50% samples from all chains).

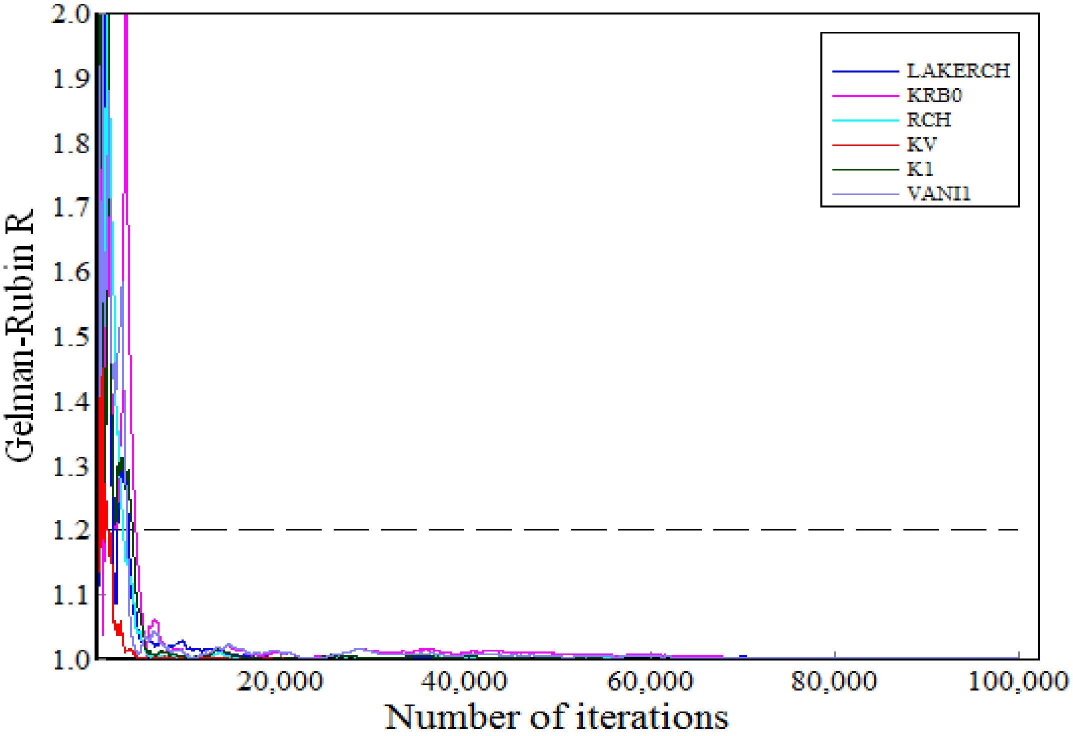


Figure D- 7. Plot of Gelman-Rubin R statistics of six model parameters. The threshold of R value 1.2 is indicated by the black dash lines.

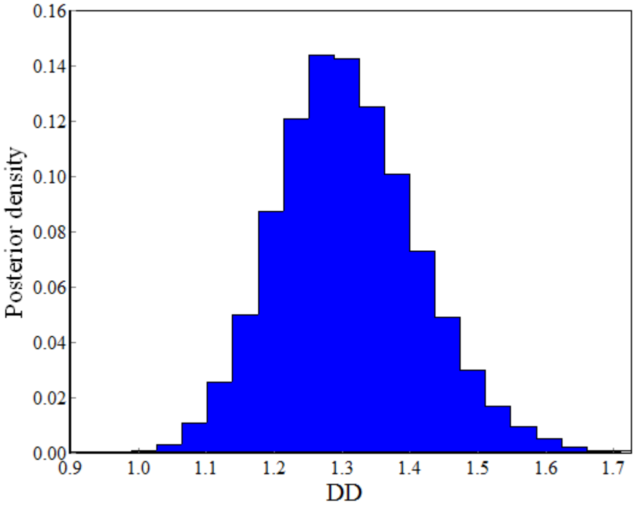


Figure D- 8. Simulated posterior density of predicted drawdown (DD).

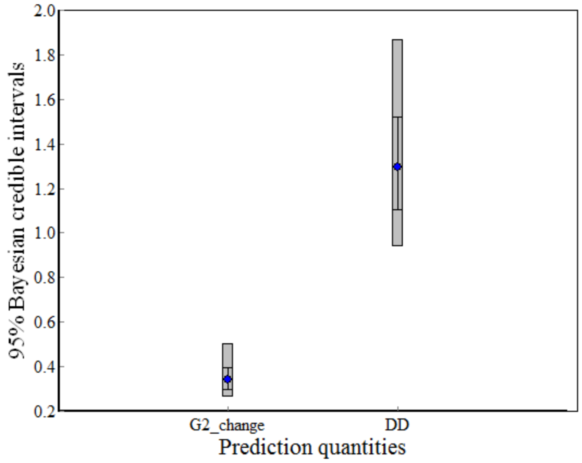


Figure D- 9. 95% Bayesian credible intervals (black line) of predicted flow change at G2 (G2_change) and drawdown (DD). The mean values are represented by blue dots and the ranges of the simulated predictions are represented by the gray bar.

Batch files for the MCMC example

The batch files distributed to run the MCMC example are listed in Tables D-4 and D-5.

Table D- 4. The batch files distributed in test-MCMC-win subdirectory HO1/Homodel_test1. [Running these batch files produces parameter and prediction samples of test example HO1 described in Appendix D.]

Batch file name	Purpose
00-a-clean.bat	Delete all output files of MCMC run
00-a-Run-forward.bat	Execute forward model using starting parameter values
To run MCMC in parallel, execute 01 to 04 in order. To run MCMC sequentially, the first two batch files do not need to be executed.	
01-runners-populate.bat	Create two runner directories
02-start-runners.bat	Start the runners. In the batch file, set N to the number of runners. The batch file is distributed with N=2.
03-RunMCMC-forward.bat	Execute UCODE_2014 MCMC mode for parameter samples
04-RunMCMC-prediction.bat	Execute UCODE_2014 MCMC_prediction mode for prediction samples.
05-RunMCMC-forward&prediction.bat	Execute UCODE_2014 MCMC and MCMC_prediction modes for parameter and prediction samples. If the runner directories created by 01 and 02 above have been created and the computer is multicore, the run will be in parallel.

Table D- 5. Comments about selected files from the MCMC examples

Selected Test Case 1 Files – in directory test-win/ chap5-test-MCMC/HO1/HOmodel_test1	
UCODE output files used as MCMC input files	
fn._paopt fn._mv	If flag ‘UseRegResult=yes’ in the MCMC_Controls input block, then the calibration results can be used to start MCMC from a multinormal distribution with _paopt as mean and _mv as covariance matrix.
fn._wt	If flag ‘Use_wt=yes’ in the UCODE_Control_Data input block, then the weights in _wt will be used in the likelihood function calculation.
MCMC main input files	
fn_MCMC-forward.in	Used to run MCMC in forward mode generating parameter samples.
fn_MCMC-pred.in	Used to run MCMC in prediction mode generating prediction samples.
MCMC main output files	
fn.#umcmc	Output of MCMC run in forward mode generating parameter samples.
fn.#umcmc_pred	Output of MCMC run in prediction mode generating prediction samples.
Other MCMC Output files	
fn._mcmc_par**	Save parameter samples generated and the associated log likelihood function values from chain **, where ** is set to 01, 02, and so on.
fn._mcmc_sim**	Save simulated values corresponding to generated parameter samples from chain **.
fn._mcmc_grr	Gelman-Rubin statistic used to diagnose simulation convergence.
fn._mcmc_restart	Last generated samples and the associated log likelihood function values from this simulation of all chains to use to restart MCMC.
fn._mcmc_pred**	Save prediction values calculated from parameter samples of chain **.
Selected Test Case 2 Files – in directory test-win/HOmodel_test2	
Other MCMC input files	
covmatrix.dat	The covariance matrix is needed when using a multinormal prior distribution. The file name will be specified in MCMC_Prior_Groups input block. If none of the parameters has multinormal prior distribution, the covariance matrix is not needed.

References

Doherty, J., 2010, PEST, Model-independent parameter estimation—User manual (5th ed., with slight additions) and addendum: Brisbane, Australia, Watermark Numerical Computing. http://www.sspa.com/PEST/index.html

Hill, M.C., Cooley, R.L., and Pollock, D.W., 1998, A controlled experiment in ground-water flow model calibration: Ground Water, vol. 36, no. 3, p.520-535.

Fienen, M.N., D’Oria, Marco, Doherty, J.E., and Hunt, R.J., 2013, Approaches in highly parameterized inversion: bgaPEST, a Bayesian geostatistical approach implementation with PEST—Documentation and instructions: U.S. Geological Survey Techniques and Methods, book 7, section C9, 86 p., available online at http://pubs.usgs.gov/tm/07/c09

Lu, D., M. Ye, and M. C. Hill (2012), Analysis of regression confidence intervals and Bayesian credible intervals for uncertainty quantification, Water Resour. Res., 48, W09521, doi:10.1029/2011WR011289.

Welter, D.E., Doherty, J.E., Hunt, R.J., Muffels, C.T., Tonkin, M.J., and Schreüder, W.A., 2012, Approaches in highly parameterized inversion: PEST++, a Parameter ESTimation code optimized for large environmental models. U.S. Geological Survey Techniques and Methods, book 7, section C5, 47 p. http://wi.water.usgs.gov/models/pestplusplus/

Appendix E: Advanced Topics for Singular Value Decomposition (SVD)

This appendix discusses three topics mentioned in Chapter 4. These include how the SVD identifiability (ID) statistic is related to the composite scaled sensitivity (CSS), the calculation of sensitivities relative to SVD parameters, and using CSS and PCC to determine which process-model parameters will have values that are unchanged when SVDupdate=Option2_css_pcc.

Relating the SVD ID statistic and CSS

This section includes two higher-level topics that require more difficult mathematics to describe. Most users will not require this level of detail so it is provided in an appendix.

CSS can be related to the singular vectors of equation 5, as follows. Pre-multiply each side of equation 5 by its transpose. **U** contains scaled orthogonal vectors, so $\mathbf{U}^T\mathbf{U}=\mathbf{I}$, where **I** is the NP-dimensional identity matrix. Thus, the following is derived:

$$\mathbf{B}(\mathbf{X}^T\boldsymbol{\omega}\ \mathbf{X})\mathbf{B} = \mathbf{D}\mathbf{S}^2\mathbf{D}^T \qquad \text{(E-1)}$$

where **X**, **B**, **V**, and **S** are defined for equations 4 and 5. Multiplication by **B** and **ω** produces the same scaling used to obtain DSS in Hill and Tiedeman (2007, eq. 4.4). Using the rules of matrix multiplication, CSS (Hill and Tiedeman, 2007, eq. 4.6) is related to equation (E-1) as follows. Divide the jth diagonal term from the left side of equation 13 by ND and take the square root to produce the following expression for CSS_j.

$$CSS_j = \{(1/ND)[\mathbf{B}(\mathbf{X}^T\boldsymbol{\omega}\ \mathbf{X})\mathbf{B}]_{jj}\}^{1/2} \qquad j=1,ND \qquad \text{(E-2a)}$$

Applying equation (13) yields:

$$CSS_j = \{(1/ND)[\mathbf{B}(\mathbf{X}^T\boldsymbol{\omega}\ \mathbf{X})\mathbf{B}]_{jj}\}^{1/2} = \{(1/ND)[\mathbf{D}\mathbf{S}^2\mathbf{D}^T]_{jj}\}^{1/2}\ j=1,ND \qquad \text{(E-2b)}$$

Using equations E-1 and E-2 and the definition of $d_{j,k}$ from equation 2, CSS can be calculated using singular values and vectors as:

$$CSS_j = \{[s_1^2(d_{j,1})^2+s_2^2\ (d_{j,2})^2 +\ldots\ s_{NPI}^2\ (d_{j,NPI})^2 +\ldots s_{NP}^2\ (d_{j,NP})^2\ ]/ND\}^{1/2}$$

$$j=1,NP \qquad \text{(E-3)}$$

From Table 2, ID_j is calculated using the first NPI terms as

$$ID_j = \{[(d_{j,1})^2+(d_{j,2})^2 +\ldots\ (d_{j,NPI})^2]\}^{1/2} \quad j=1,NP \qquad \text{(E-4)}$$

Here the summation is listed in a way that clearly shows NPI terms of the singular vectors used to define the ID statistic.

Often both CSS and ID are presented as bar charts. Under what conditions would one expect the bars to be the same height such that the two graphs could be used interchangeably to identify important and unimportant process-model parameters?

When NPI=NSVD, which is one alternative proposed by Doherty and Hunt (2009), the contribution to CSS from the terms related to unestimated (null space) SVD parameters tend to be small because the singular values associated with these vectors are small. Indeed, singular value s_{NSVD} is often 5 to 6 orders of magnitude smaller than s_1, and subsequent values are even smaller. Thus, when NPI=NSVD, the latter terms in equation (15) (terms with the second subscript on d equal to NSVD+1 to NP) are generally minuscule. However, a bar chart of ID statistics can differ substantially from a bar chart of CSS because of the range in singular values within the estimated SVD parameters.

The CSS values and the fractional contribution of each vector element to each CSS is provided by UCODE_2014 in the _sc_svd file.

The ID statistics measure the representation of the process-model parameters in NPI of the SVD parameters with the NPI largest singular values. This is closely related to the parameter importance measured by a CSS/PCC analysis, but is not the same. In part this is because the ID statistic depends on the value of NPI considered. Often ID values calculated for values of NPI that are smaller than NSVD are similar to graphs of CSS. It is sometimes instructive to consider ID statistics for a range of NPI, which was also suggested by Doherty and Hunt (2010a). ID statistics for NPI=NSVD are output to the data-exchange file _svd-id. ID statistics for other values of NPI can be calculated from the singular vectors listed in data_exchange file _svd, and by using GW_Chart.

In general, parameters with larger CSS values tend to be represented more in the estimated SVD parameters. Parameters with high correlation (absolute value of PCC close to 1.0) are represented in the estimated SVD parameters less than their sensitivities might indicate.

The ID statistics can be plotted using bars that distinguish contributions shown in the summations of equations E-4; an example stacked graph for SVD is shown in Figure 8A. Figure 8A is stacked using the component values in equation E-4, as suggested by Doherty and Hunt (2009). Similar graphs for CSS constructed based on equation E-3 are shown in Hill (2010).

Calculation of SVD sensitivities

As noted in Chapter 3, SVD parameters need to be converted to process-model parameters when performing process model runs to calculate perturbation sensitivities of the SVD parameters. Sensitivities of the SVD parameters are needed in the parameter estimation process. In UCODE, the SVD parameter sensitivities are calculated by perturbation (forward, central, or backward perturbation) as illustrated using forward differences in equation 9. The sensitivities are calculated as follows, where **a** is used for SVD parameters and **b** is used for process-model parameters. In general, when regression is conducted using SVD parameters, the Δ**a** vectors include all zeros except one term, while the Δ**b** vectors include as many non-zero terms as there are non-zero coefficients in equation 2.

$$\frac{\Delta y'}{\Delta a} = \frac{y'(\mathbf{a} + \Delta\mathbf{a}) - y'(\mathbf{a})}{(\mathbf{a} + \Delta\mathbf{a}) - (\mathbf{a})} = \frac{y'(\mathbf{b} + \Delta\mathbf{b}) - y'(\mathbf{b})}{(\mathbf{b} + \Delta\mathbf{b}) - (\mathbf{b})} \qquad \text{(E-5)}$$

where:
a value of an SVD parameter for which sensitivities are calculated;

y'(**a**) the value of a simulated equivalent to an observation, y, calculated using the SVD parameter values in vector **a**. To run the process model, the SVD parameters in **a** need to be converted to the process-model parameters identified as **b** using equation 8.

a a vector of SVD parameter values that are originally calculated using starting process-model parameters and equation 3, then updated by the parameter estimation process;

b a vector (can be thought of as a list) of process-model parameter values calculated using the SVD parameter values in **a** and equation 8;

Δ**a** a vector of SVD parameter changes in which all values are zero except the value associated with the SVD parameter for which sensitivities are being calculated;

Δ**b** a vector of process-model parameter changes consistent with the SVD parameter change in Δa. The number of non-zero elements in Δ**b** generally equals the number of non-zero coefficients in equation 2.

y'(**a** + Δ**a**) the value of y' calculated using the process-model parameter values associated with **a** + Δ**a**;

Δy' the change in y' caused by the SVD parameter value changes in Δ**a**.

For parameter estimation, the simulated equivalents represented by y' can be observations, prior information, or regularization. For observations, values are simulated using the process model. For prior information and regularization, the changed process-model parameters are used in the prior information equations to obtain y'(**b** + Δ**b**). However, for prior information and regularization that involve linear expressions, the sensitivities can be calculated once using equation 2; that is, $\mathbf{A}^T\mathbf{b}=\mathbf{D}$. Each row of the process-model parameter sensitivity array **X** related to prior information and regularization generally has one element for each process-model parameter. These lines can be transformed to SVD parameter space as $\mathbf{A}^T(\mathbf{x}_i^{PM})^T=(\mathbf{x}_i^{SVD})^T$, where i identifies an item of prior information or regularization, superscript PM stands for process model, and superscript SVD stands for SVD model.

UCODE_2014 Performance with SVDupdate=Option2_css_pcc

Chapter 4 discussed the alternative of setting the values of NPnull process-model parameters instead of setting the values of NPnull SVD parameter values. This alternative is accessed using two of the three alternatives for the SVDupdate keyword in the UCODE_Control_Data input block. The advantage of this approach is that the NPnull process-model parameters that are least informed by the observations and prior information do not change in value, which makes it more obvious to the modeler and users of the model which process-model parameters are not important to model calibration. When using this approach, the user must determine which NPnull process-model parameter values to set.

One option is to use the singular vectors to identify the NPnull process-model parameters. In one alternative, the NPnull process-model parameters with the smallest ID values when evaluated for NPI=NSVD would be set. Consider this option using results from Hill and Østerby (2003), which are shown in Figure E-1. In this problem, all six parameters except ANIV form an correlated group.

The correlation results because these five parameters can all be multiplied or divided by an arbitrary number and the same hydraulic heads are produced; for this problem all the observations are heads. The very small sixth singular value (4.3×10^{-6}) and the relatively large values of the five elements in singular vector 6 that are not associated with parameter ANIV reflect the correlation. A necessary (though possibly not sufficient) requirement for obtaining a tractable regression problem is that one of the parameters in the set of 5 correlated parameters needs to be set.

Consider choosing the parameter using ID. With the smallest ID value for NPI=5, the largest number of parameters that can be estimated based on the singular values and an SVDratio of less than 1×10^{-3}. The ID graph is shown in Figure E-1b. The parameter chosen through this strategy would be K1. Inspection of the CSS values shows this is the most sensitive of the correlated parameters. Hill and Østerby (2003) suggest that omitting K1 would reduce the correlation the most. However, they also note that “The dominant consideration is rarely the ability to achieve the maximum reduction in correlation because even a small reduction often is sufficient to obtain unique parameter estimates.” Setting the value of the parameter for which the observations provide plentiful information, as measured by CSS, is not likely to be advantageous to model development. Alternatively, choosing one or more parameters with large PCC and small CSS is a way to allow parameters better informed by the observations to be estimated.

The strategy used in UCODE_2014 is either to allow the user to define the order of set process-model parameters with SVDset, as described in Chapter 4, or to use PCC and CSS to identify a set composed of selected correlated and insensitive parameters to retain their starting values. The logic followed by UCODE when using CSS and PCC is described in chapter 4. Here we show how the method would perform for the problem described in Figure E-1.

(a)

Singular values

σ_1	σ_2	σ_3	σ_4	σ_5	σ_6
2.2	0.96	0.38	0.19	0.02	4.3E-06

Singlar vectors

	v_1	v_2	v_3	v_4	v_5	v_6	CSS	Ratio
K1	0.45	-0.15	0.017	-0.36	0.35	-0.72	43	1.00
KRB	0.42	0.26	0.76	0.43	-0.018	-0.0038	0.22	0.005
ANIV	-0.22	-0.90	0.36	0.12	0.0008	2E-06	0.60	0.014
K_{2M}	0.49	-0.22	-0.40	0.40	-0.64	-0.20	12	0.279
RCH1	-0.44	0.13	-0.16	0.65	0.31	-0.48	29	0.674
RCH2	-0.44	0.20	0.32	-0.29	-0.61	-0.45	27	0.628

PCC

	K1	KRB	ANIV	K_{2M}	RCH1	RCH2
K1		1.00	-0.31	1.00	1.00	1.00
KRB			-0.31	1.00	1.00	1.00
ANIV				-0.31	-0.31	-0.31
K_{2M}					1.00	1.00
RCH1						1.00
RCH2						

(b)

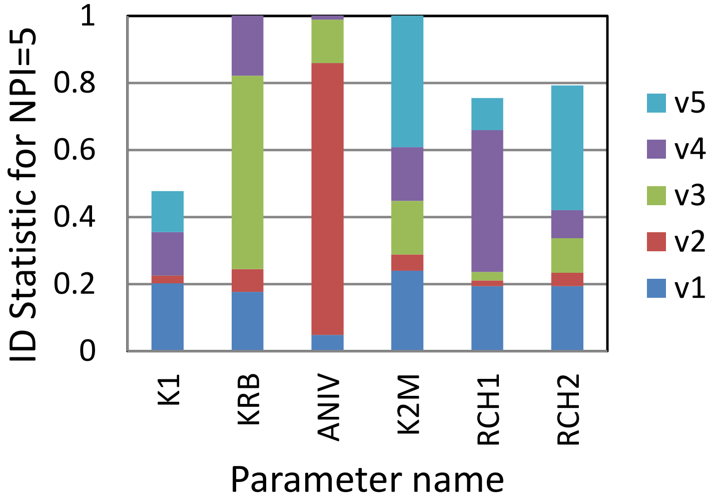


Figure E- 1. Results from Hill and Osterby (2003, Tables 2, 5, and 10) for a problem with all but one of the six parameters being extremely correlated. (a) The singular values and vectors and CSS and PCC. (b) The ID statistic with NPI=5.

Using the method described in chapter 4 for when SVDupdate=Option2_css_pcc, with NSVD=5, NPnull = 1, yields the following NPnull parameter. Default values are used for the other keywords; that is, SVDupdate_CSSrat=0.02, SVDupdate_CSSabs=1.0, and SVDupdate_CSSabs=1.00, and SVDupdate_PCC=0.97.

Step 1: Set parameter KRB because of the small CSS ratio. This provides the one parameter needed (NPnull=1) so steps 2 through 4 are not needed.

KRB is one of the correlated parameters, so selecting KRB also satisfies the need to set one of the correlated parameters. If desired, this could be assured by proceeding directly to step 3 by setting SVDupdate_CSSrat = SVDupdate_CSSabs = 0.0.

The results of Hill and Østerby (2003) suggest that PCC is susceptible to numerical problems that can yield a smaller PCC value than would be consistent with actual parameter dependence. This means that the method for using PCC to detect parameter dependence is approximate. In some circumstances values less than 0.97 are needed to capture dependent parameters. This is why the SVDupdate_PCC keyword of the Parameter_Data input block allows values other than the default value of 0.97 to be specified. Hill and Østerby (2003) suggest that the problems with calculating PCC are more serious for parameters with small sensitivities. Small sensitivity parameters are more common as the number of parameters increases. The method descrived in this work tends to work better for models parameterized with relatively few parameters.

References

Doherty, J., R.J. Hunt, 2009, Two statistics for evaluating parameter identifiability and error reduction. Journal of Hydrology 366, 119–127.

Doherty, J.E., and Hunt, R.J., 2010, Approaches to highly parameterized inversion—A guide to using PEST for groundwater-model calibration: U.S. Geological Survey Scientific Investigations Report 2010–5169, 59 p.

Hill, M.C., 2010, Comment on ‘‘Two statistics for evaluating parameter identifiability and error reduction” by John Doherty and Randall J. Hunt: Journal of Hydrology, 380, p. 481-488.

Hill, M.C. and Østerby, Ole, 2003, Determining extreme parameter correlation in ground-water models: Ground Water, v. 41, No. 4, p. 420-430.

.

Appendix F: Calculations and Data-Exchange Files Behind the Stacked CSS Graphs

Division of CSS into observation components as shown in Figure 8a uses the _sc and the _scgrp data-exchange files produced by UCODE_2014. The _sc and _scgrp files used to produce Figure 8a are shown in Figures F-1 and F-2. The CSS values provided in the _sc file are referred to in the equations below as CSS_j, where subscript j identifies a parameter.

The _scgrp file provides CSS value for observations groups that are appropriate for graphs of CSS for each group individually, as shown by Barlebo et al. (1998). Here these values are labeled $CSSgroup_{j,k}$, where j identifies the parameter and k identifies and observation group. The values in the _scgrp files are sometimes larger than CSS_j in the _sc file because the equation for CSS (Hill and Tiedeman, 2007, p. 50) has a division by the number of observations prior to taking the square root. Here, the number of observations in each group is labeled $ngroup_k$. In this problem the total number of observations is 119; $ngroup_k$ ranges between 5 and 50.

The stacking shown in Figure 8a is a useful way to depict observation contribution. For this stacking, the contributions for each parameter j and observation group k, $CSSstack_{j,k}$, are calculated using values of $CSSgroup_{j,k}$ and $ngroup_{,k}$ from _scgrp and values of CSS_{all} from _sc as follows.

$$CSSstack_{j,k} = (CSSgroup_{j,k} \times ngroup_k^{1/2}) \times$$

$$[CSS_j/\{\Sigma_{k=1,\#groups}(CSSgroup_{j,k} \times n_{group\ k}^{1/2})\}] \qquad \text{(F-1)}$$

Where #groups is the number of observations groups. The first term to the right of the equals sign removes the division by the number observations in each group. The second term normalizes the values so they add up to CSS_j.

```
“PARAMETER NAME” “COMPOSITE SCALED SENSITIVITY”
 T110              0.230824007826494
 T70                4.74650673911119
 T30                4.25309941280302
 T16                9.70248643868573
 T08                2.06545225733053
 P110              0.309197545609988E-03
 P70                3.63133864700400
 P30                2.39422699593356
 P16               0.582305009802810
 P08               0.199050530484271
```

Figure F- 1. The first two columns of the _sc file. These composite scales sensitivity values are labeled CSS_j in equation F-1.

```
“GROUP NAME” headobs       “NUMBER IN GROUP”         15
 “PARAMETER NAME” “COMPOSITE SCALED SENSITIVITY”
 T110              0.592036340956502E-01
 T70                10.0112975190524
 T30                6.64655649861694
 T16                10.9297315690082
 T08                1.51016342455099
 P110              0.266757352931017E-04
 P70               0.229418400113439E-04
 P30               0.345196359624527E-04
 P16               0.295940898430284E-04
 P08               0.490762236233461E-04
 “GROUP NAME” xyzobs        “NUMBER IN GROUP”          5
 “PARAMETER NAME” “COMPOSITE SCALED SENSITIVITY”
 T110              0.300805640119536E-03
 T70               0.459560676120824E-02
 T30               0.243780813861019
 T16               0.430372347869675E-02
 T08               0.198453642644289
 P110               0.00000000000000
 P70                0.00000000000000
 P30               0.417869731069874E-02
 P16               0.237364406042714E-02
 P08               0.863913633388704E-03
“GROUP NAME” timobs        “NUMBER IN GROUP”         14
 “PARAMETER NAME” “COMPOSITE SCALED SENSITIVITY”
 T110              0.318202456702838
 T70                2.08428710689089
 T30                2.20477929481704
 T16                21.7727359706001
 T08                2.66746312208907
 P110              0.671362354129866E-03
 P70                1.24288556715714
 P30               0.643631632017602
 P16               0.894225188054247
 P08               0.544099854517366
“GROUP NAME” cncobs        “NUMBER IN GROUP”         50
 “PARAMETER NAME” “COMPOSITE SCALED SENSITIVITY”
 T110              0.294529251759959
 T70                3.34209348448956
 T30                4.11393254728170
 T16                5.21248438887020
 T08                2.08944306784850
 P110              0.286352323218257E-03
 P70                4.73613705689189
 P30                3.10537268863596
 P16               0.435235541748668
 P08               0.590038134901739E-01
```

Figure F- 2. Selected columns of the _scgrp data-exchange file from the test problem presented in Chapter 4. These composite scaled sensitivity values are labeled CSSgroup,k in equation 18, where k ranges from 1 to #groups. Here, #groups equals 5.

```
   "GROUP NAME" typobs        "NUMBER IN GROUP"         25
    "PARAMETER NAME" "COMPOSITE SCALED SENSITIVITY"
    T110               0.00000000000000
    T70                3.64969204876881
    T30                3.97220610626605
    T16                4.35410061302181
    T08                2.12315489219167
    P110               0.00000000000000
    P70                3.43024900374141
    P30                2.33971339610824
    P16               0.807295631666080
    P08                0.00000000000000
```

Figure F- 2 – continued: Selected columns of the _scgrp data-exchange file from the test problem presented in Chapter 4. These composite scaled sensitivity values are labeled CSSgroup,k in equation 18, where k ranges from 1 to #groups. Here, #groups equals 5.

Computational Details

Equation F-1 can be derived from the eqution for CSS (eq. 4.6 of Hill and Tiedeman, 2007) as follows. First, consider the following equation for the square of CSS.:

$$CSS_j^2 = 1/N\{[\Sigma_{i=1,N1}\ (dss_{ij})^2|_{\mathbf{p}}] + [\Sigma_{i=1,N2}\ (dss_{ij})^2|_{\mathbf{p}}] + \ldots$$
$$+[\Sigma_{i=1,Nn}\ (dss_{ij})^2|_{\mathbf{p}}]\} \quad j=1,NP \qquad \text{(F-2)}$$

To calculate contributions to the CSS values, instead of CSS^2, two options are possible.

1. Calculate proportional contributions as defined for CSS^2 (eq. F-2) and use those proportions to define the stacking for Figures such as that shown in Figure 8a.
2. Take the square root of each term in equation F-2 and use them to calculate the proportions used to define the stacking.

Because the final values are often less than 1.0, the latter allows observations groups with small contributions to decline in relative value more slowly, which means that in more circumstances they remain visible when plotted as stacked bar charts. For this reason, here the second option is used to construct Figure 8a. For this stacking, the contributions for each parameter j and group k, $CSSstack_{j,k}$, are calculated using values of $CSSgroup_{j,k}$ and $ngroup_{,k}$ from _scgrp and values of CSS_{all} from _sc as shown in equation F-1.

Reference

Hill, M.C. and Tiedeman, C.R., 2007, Effective groundwater model calibration, with analysis of sensitivities, predictions, and uncertainty: Wiley and Sons, New York, New York, 455 p.

Appendix G: Comparison of UCODE_2014 with UCODE_2005, PEST, and OSTRICH

The aspects of UCODE_2014 that differ from UCODE_2005 are explained in this appendix. Differences between the implementation of SVD in UCODE_2014 relative to its implementation in PEST (Doherty, 2010) are also noted. UCODE and PEST are largely gradient based methods while OSTRICH uses heuristic methods like genetic algorithms.

Table G- 1. Comparison of UCODE_2014 and UCODE_2005 as described in Poeter et al. (2005).

[New UCODE_2014 input blocks are described in Chapters of this report. All input blocks are included in the abbreviated input instructions of Appendix F of this report.]

Capabilities that are new in UCODE_2014 relative to UCODE_2005
_For_Prediction input blocks are included to support parameters required only in the simulation of predictions Enhanced support of weighting so that coefficients of variation can be used to calculate variances using simulated values. SVD parameter transformation to assist in estimating parameters Markov Chain Monte Carlo capability using the DREAM algorithm that supports identification of multiple minima.
SVD capability in UCODE_2014 relative to PEST
Set parameter values. UCODE_2014 allows the user to define process-based parameters that are not allowed to change during the SVD optimization using SVDupdate=Option2_SVDset or Option2_css_pcc. Normally in SVD, the values of SVD parameters are set, and all process-model parameter values change. The process-model parameters defined to be set with this capability of UCODE_2014 are insensitive parameters and selected highly correlated parameters. **Highly parameterized problems** (in general, NP>NOBS). PEST allows NP to be greater than NOBS, and thus simulate highly parameterized problems. **Determining the number of estimated SVD parameters** (NS). PEST allows the user to define the sum of squared weighted residuals for observations and determine NS based on the specified value.
Comparison of UCODE_2014 and OSTRICH (Matott, 2011)
Optimization algorithm. Ostrich uses heuristic approaches such as genetic algorithms. **Management Optimization.** OSTRICH supports management optimization of variables involved in managing simulated systems.

Table G- 1. Comparison of UCODE_2005 and PEST as of February, 2010

Capability	UCODE_2005	PEST
Observations		
Weighting	Input as statistics (standard deviation, coefficient of variation, or variance) or weights. For coefficient of variation, convert to standard deviation using observed or simulated values.	Input as weights. What PEST calls weights is the square root of what UCODE calls weights.
Non-detect observations (Concentration, head below land surface)	Nondetect keyword in Observation input blocks.	--
Reading values	PEST methods plus a simple standardfile option to read data in columns.	PEST methods.
Parameters		
Pilot points	Limited support using MODFLOW additive parameter capability. Can use PEST utilities but optimization algorithms not as effective for large number of parameters.	Thorough support with MODFLOW using PEST utilities. Optimization methods customized for large number (thousands and more) of parameters.
SVD	To estimate parameters.	To estimate parameters Number of parameters can be greater than number of observations.
Other interpolation	As supported by model (MODFLOW supports additive parameters so any linear interpolation)	
Sensitivity Calculation		
Perturbation sensitivities	Forward, backward, and central differences. Perturbation calculated as defined fraction of parameter value.	As for UCODE_2014 and can define perturbation as an absolute quantity and change to central difference during run.
Model calculated sensitivities	Read from file produced by the model.	
Sensitivity Analysis (SA)		
Observation-Parameter SA[1]	CSS/PCC,	Composite sensitivity[2],
	Stacked CSS	--
	CSS-SVD, ID	SVD Parameter ID[4]
	OPA[3],	Relative error statistic,
	Leverage, Cook's D, DFBETAS	Leverage, Cook's D, DFBETAS
Parameter-Prediction SA[1]	PSS,	--
	PPR[3]	Predvar2-4/Predunc2-4
Observation-Prediction SA[1]	OPR[3]	Predvar1,5/Predunc1,5
Regression		
Marquardt parameter	Added after matrix is scaled	Added without scaling matrix

Capability	UCODE_2005	PEST
Line search for minima	--	Performed to calculate nonlinear confidence intervals.
Model Discrimination Criteria	SOSWR, CEV, standard error, AIC, AICc, BIC, KIC (same for both codes)	
Uncertainty Analysis		
Linear intervals on parameters	Individual and simultaneous Confidence	Individual Confidence
Linear intervals on predictions	Individual and simultaneous Confidence and Prediction	Individual Confidence
Nonlinear intervals on parameters	Individual and simultaneous Confidence and Prediction	Individual Confidence
Nonlinear intervals on predictions	Individual and simultaneous Confidence and Prediction	Individual Confidence[5]
Multi-Model Analysis: to account for conceptual model uncertainty	**MMA** (Poeter and Hill, 2007): calculated model probabilities and model-averaged parameter estimates, predictions, and confidence and prediction intervals. Allows common model discrimination criteria and new ideas to be considered.	--

[1] SA, sensitivity analysis. Similar statistics for each model are placed on the same line.

[2] Cannot be used directly to compare the relative sensitivity of different parameters. CSS=The PEST composite sensitivity × the parameter value × the square root of the number of observations.

[3] OPA, PPR, and OPR are calculated by the program OPR-PPR (Tonkin et al., 2007), for which most input files are produced by running UCODE_2014.

[4] SVD parameter identifiability statistic. Measures how the estimated SVD parameters are constituted from the process-model parameters.

[5] In the Prediction Analyzer

References

Doherty, J., 2010, PEST, Model-independent parameter estimation—User manual (5th ed., with slight additions) and addendum: Brisbane, Australia, Watermark Numerical Computing. http://www.sspa.com/PEST/index.html

Matott, L. S., 2010, OSTRICH: An Optimization Software Tool. Accessed 12/27/2010 at http://www.civil.uwaterloo.ca/lsmatott/Ostrich/OstrichMain.html

Poeter, E.P., Hill, M.C., and Banta, E.B., 2005, UCODE_2005 and six other computer codes for universal sensitivity analysis, calibration, and uncertainty evaluation: U.S. Geological Survey Techniques and Methods, book 6, sec. A, chap. 11, 283 p. http://igwmc.mines.edu/freeware/ucode/

Appendix H: Useful Mathematical and Statistical Web Sites

While this document, Hill and Tiedeman (2007), Aster et al. (2013) and other works referenced here form a useful basis for understanding methods discussed, some readers will benefit from more fundamental treatments. Here are a few web sites that may be useful. The reader is also encouraged to search the web for additional sites.

Matrix Index Notations

The following web sites are useful for readers not familiar with the matrix notation used in Chapter 4.

http://www.purplemath.com/modules/matrices2.htm

http://en.wikipedia.org/wiki/Matrix_multiplication

Linear and Nonlinear Regression

http://en.wikipedia.org/wiki/Regression_analysis

Appendix I: Running UCODE_2014 and Abbreviated Input Instructions

Input Block List

In the UCODE main input file, the input blocks need to appear in a certain order. Table I-1 displays the needed order. The input blocks printed in italics are described in this documentation. New keywords for some other input blocks also are described in this documentation. For all other input instructions, please see the UCODE_2005 documentation.

Running UCODE_2014 and Auxiliary Codes

The Run Command for UCODE_2014 needs to be executed from the directory containing the UCODE_2014 input files. The process-model input, output, and batch files can be in one or more separate directories. The process model(s) need to execute completely from one batch file without human intervention. The batch file can in turn run other batch files, and in this way there can be multiple process models.

The UCODE_2014 run command is of the form:

```
path:\UCODE_2014     input-file     fn     [fn2]
```

where:

path:\ the relative or absolute path to the UCODE_2014.exe on your computer (Alternatively you could specify this in your system path variable) .

input-file the name of the main UCODE_2014 input file (These files have extension ‘in’ in the examples distributed; see appendices C and D. An abbreviated description of input for this file is provided in this appendix).

fn filename prefix for UCODE_2014 output files Spaces are not allowed in fn, even on operating systems that allow spaces in filenames.

fn2 an optional filename prefix for UCODE_2014 prediction mode files. Spaces are not allowed in fn2, even on operating systems that allow spaces in filenames. If fn2 is not present then this root is set to fn, thus making UCODE_2014 back compatible with previous versions of UCODE. Output files that use fn2 are produced by three modes of UCODE_2014 and three auxiliary codes (LINEAR_UNDERTAINTY, CORFAC, and MODEL_LINEARITY_ADV). The associated filename suffices are as follows:

from a UCODE_2014 prediction run: dmp, mvp, p, prp, pv, sppp, sppr, spsp, spsr, spu, suprip, wtprip, gmp, paoptp, #upred

from a LINEAR_UNCERTAINTY run: linp, #linunc

from a CORFAC run: cfsu, cfconf, cfpred, b3conf, b3pred, b1advconf, b1advpred, #corfac_conf, #corfac_pred

Table I- 1. Blocklabels of the main input file for UCODE_2005, including blocklabels for the new input blocks described in this report. Modified from Poeter et al. (2005, Table 4, p. 43).

[Bold type and grey shading: required input blocks (except as described in footnote [2]); other input blocks are optional. Italic type: new input blocks described in this document.]

Purpose	**Blocklabel**	**Default column order[1]**
Define UCODE operation	Options	No
	Merge_Files	No
	UCODE_CONTROL_DATA	No
	MCMC_Controls	No
	Reg_GN_Controls	No
	Reg_GN_NonLinInt	No
	Model_Command_Lines	No
Define parameters	Parameter_Groups	No
	Parameter_Groups_For_Prediction	No
	Parameter_Data[2]	Yes
	Parameter_Data_For_Prediction	Yes
	Parameter_Values	Yes
	Parameter_Values_For_Prediction	Yes
	Derived_Parameters	Yes
	Derived_Parameters_For_Prediction	Yes
	MCMC_Prior_Groups	Yes
	MCMC_Prior_PDF[2]	Yes
Define observations	Observation_Groups	No
	Observation_Data[3]	Yes
	Derived_Observations	Yes
Define predictions	Prediction_Groups	No
	Prediction _Data[3]	Yes
	Derived_ Predictions	Yes
Define prior information	Prior_Information_Groups	No
	Prior_Information_Groups_For_Prediction	No
	Linear_Prior_Information	Yes
	Linear_Prior_Information_For_Prediction	Yes
Define variance-covariance matrices to weight groups of observations or prior information with correlated errors.	Matrix_Files	No
Interact with process-model input and output files.	**Model_Input_Files**	Yes
	Model_Output_Files	Yes
Run process model(s) using multiple processors	Parallel_Control	No
	Parallel_Runners	Yes

[1] 'Yes': the input block has a default column order. With blockformat TABLE, these blocks can contain data without column labels for selected keywords if the data are in default order. Keywords defined in the JUPITER API are supported. Keywords added for UCODE always need column labels.

[2] The Parameter_Data input block is needed when MCMC=no by designation or default in the UCODE_Control_Data input block. The MCMC_Prior_PDF input block is needed when MCMC=yes.

[3] Often either the Observation_Data or the Prediction_Data input block is required.

from a UCODE_2014 model linearity advanced run: #umodlinadv_conf, #umodlinadv_pred

from a MODEL_LINEARITY_ADV run: #modlinadv, #modlinadv_conf, #modlinadv_pred

from a UCODE_2014 non-linear uncertainty run: #unonlinint_conf, #unonlinint_pred, intconf, intpred, intconf_wr, intpred_wr, intconf_sum, intpred_sum, intconf_par, intpred_par, DATE-intconf, DATE-intpred

Instructions for running most auxiliary codes have not changed and are described in Chapter 4 of the UCODE_2005 documentation (Poeter et al., 2005). The one change is that a new optional argument has been added to the command line of auxiliary code RESIDUAL_ANALYSIS to allow analysis of model fit from the beginning of an SVD run. The new RESIDUAL_ANALYSIS run command is of the form:

```
path:\ RESIDUAL_ANALYSIS     fn    [presvd]
```

where:

path:\ the relative or absolute path to RESIDUAL_ANALYSIS.exe on your computer (Alternatively you could specify this in your system path variable).

fn filename prefix used for the UCODE_2014 run (Spaces are not allowed in fn, even on operating systems that allow spaces in filenames).

presdv is literally the inclusion of the letters “presvd”. If this is included on the command line, RESIDUAL_ANALYSIS will use input files with filename suffices ending in “_presvd”. RESIDUAL_ANALYSIS output files also will end with “_presvd”.

Abbreviated Input Instructions

The abbreviated input instructions listed here are intended for quick reference by experienced users. Complete input instructions are presented in Chapters 6 through 13 and 17 of Poeter et al. (2005) and in this UCODE_2014 documentation. The input blocks are presented here in the order they need to appear in the UCODE_2014 main input file. If listed as optional, the input block can be omitted, but if present needs to be in the order shown. Most input blocks are documented in Poeter et al. (2005) and the relevant chapter number of that report is preceded by “UCODE_2005”. For input blocks from UCODE_2005, keywords for which performance is different in UCODE_2014 are presented shaded with gray.

The Graphical User Interface (GUI) ModelMate, can be used to create and manage UCODE input and model runs. ModelMate can be obtained from the USGS software distribution site: http://water.usgs.gov/software/ModelMate/, and is discussed briefly in Chapter 5 of this UCODE_2014 report. ModelMate can be used with the ModelMuse GUI to create seamless support for MODFLOW2005.

When the options “yes” and “no” are listed below, “true” and “false” can be used instead.

As always in UCODE, keywords are not case sensitive.

Options Input Block (optional) UCODE_2005 Chapter 6

Verbose - Controls printing to the UCODE_2005 main output file.

Derivatives_Interface - Filename or path of file defining how to read derivatives.

PathToMergedFile - Filename or path of merged file. If the file exists, it is replaced.

Merge_Files Input Block (Optional) UCODE_2005 Chapter 6

PathToFile - Path of a file.

SkipLines - Lines to skip at the top before file is appended. Default=0.

UCODE_Control_Data Input Block (optional) UCODE_2005 Chapter 6, except as noted

ModelName - Identifies the model. Up to 12 characters. Default=generic.

ModelLengthUnits - Defines the LENGTH units. Up to 12 characters. Default=NA.

ModelMassUnits - Defines the MASS units. Up to 12 characters. Default=NA.

ModelTimeUnits - Defines the TIME units. Up to 12 characters. Default=NA.

For the following seven variables, see Table 3 of Poeter et al. (2005) for modes produced by "yes".

Sensitivities - yes: calculate sensitivities. Default is no.

Optimize - yes: estimate parameters. Default is no.

Linearity - yes: do the calculations and produce file fn._b2. Default=no.

Prediction - yes: determine predictions and their sensitivities. Default=no.

LinearityAdv - conf, pred, or no. Default=no.

NonlinearIntervals - yes: calculate nonlinear confidence intervals. Default=no.

SOSsurface - yes or file: calculate objective-function values. Default=no.

SOSfile - file used when SOSsurface=file.

StdErrOne - yes: calculate statistics with s^2 replaced by 1.0. Default=no.

EigenValues - yes: calculate eigenvalues and eigenvectors of the parameter variance-covariance matrix. Default=no.

Three keywords control printing of tables of observations, simulated values, and residuals to the main output file.

StartRes - For the starting parameter values. Default=yes.

IntermedRes - For parameter-estimation iterations. Default=no.

FinalRes - For the final parameter values. Default=yes.

Three keywords control printing of sensitivity tables to the main output file.

StartSens - For the starting parameter values. Default=dss.

IntermedSens - For parameter-estimation iterations. Default=none.

FinalSens - For the final parameter values. Default=dss.

DataExchange - yes: generate the data-exchange files. Default=yes. Now always set to yes. A designation of no is ignored.

CreateInitFiles - yes: generate only the _init data-exchange files. Default=no.

The following keywords are added for UCODE_2014

Reactivate - final, starting, or no. Default=final. Reactivate governs performance of UCODE when the regression converges, or the regression fails to converge and stats_on_nonconverge=yes. The original UCODE_2005 functionality is achieved with Reactivate=no.

Singularvalues - yes, no. Default=no. Controls printing of the singular values and singularvectors of the matrix $\mathbf{X}^{T}\boldsymbol{\omega}^{1/2}$ to the fn.#uout file. The SVD data-exchange files are always printed.

SVD - yes, no. Yes, use SVD transformed parameters in the regression. Default=no.

SVDphase - start_with_pmp (pmp refers to process-model parameters), continue_from_pmp, continue_from_presvd, continue_from_svd. Default= start_with_pmp.

SVDnumber - number of svd parameters to use, but if SVDratio is specified then this is ignored. Default is as shown for SVDratio.

SVDratio - determines SVDnumber as the number of singular values with a ratio to largest singular value greater than the value of SVDratio. Common values are between $1x10^{-5}$ and $1x10^{-6}$. Default=$1x10^{-6}$.

SVDupdate - svdall, option2_svdset, option2_css_pcc. Svdall: all process-model parameters are updated. Default=svdall.

Use_wt Use observation weighting read from the _wt file from a previous regression. Default=no.

MCMC **-** yes, no. Yes, conduct a Markov-Chain Monte Carlo simulation. Default=no.

MCMC_Prediction - yes, no. Yes, calculate predictions using the parameter values produced in a previous UCODE run with MCMC=yes. Default=no.

Reg_GN_Controls Input Block (optional) UCODE_2005 Chapter 6, except as noted

Keywords that control when parameter-estimation iterations stop.

TolPar - Tolerance based on parameter values. Default=0.01.

TolSOSC - Tolerance based on model fit. Default=0.0.

MaxIter - Maximum number of parameter-estimation iterations. Default=5.

Keywords that restrict how much parameter values can change in one parameter-estimation iteration.

MaxChange - Maximum fractional amount parameter values are allowed to change between parameter-estimation iterations. Default=2.0.

MaxChangeRealm - Indicates whether MaxChange applies in native or regression space. Default=Native.

Keywords that control the regression when SVD=yes in the UCODE_Control_Data input block.

SVDperturbamt - For all svd parameters, perturbation used to calculate sensitivities equals SVDperturbamt times the current parameter value. Default=0.01

SVDmaxchange - The maximum fractional amount any SVD parameter can change in any parameter-estimation iteration. Default=2.0.

SVDtolpar - Regression converges when all estimated SVD parameters change by less than SVDtolpar times the current parameter value. Default=0.01

Keywords used to calculate the Marquardt parameter

MqrtDirection - Angle (in degrees) between down-gradient direction on the sum-of-squared-residuals surface and the parameter update vector. Default=85.4^{o}.

MqrtFactor - See equation 8 for the Marquardt parameter. Default=1.5.

MqrtIncrement - See equation 8 for the Marquardt parameter. Default=0.001.

Keywords that control quasi-Newton updating

QuasiNewton - yes, no. Yes: use quasi-Newton updating as indicated by the criteria below. Default=no.

If either of the following two criteria is met for a parameter-estimation iteration, Quasi-Newton updating is used for that and all subsequent iterations.

QNiter - Number of iterations executed before including the Quasi-Newton enhancement. Default=5.

QNsosr - Fractional change in the sum-of-squared weighted residuals over two parameter iterations below which Quasi-Newton matrix enhancement is employed. Default=0.01.

Keywords that control omission of observations for regression based on values defined by the process model.

OmitDefault - The number of values to read from user-created file fn.omit. Default=0.

OmitWeight - The weight used for omitted observations. Default = 1E-70

Keyword that controls printing of final statistics.

Stats_On_Nonconverge - yes: calculate final sensitivities and calculate and print statistics when parameter estimation does not converge in the maximum number of iterations. Default=yes.

Keywords that control dynamic omission of insensitive parameters from regression.

OmitInsensitive - yes: omit parameter j from the regression if its composite scaled sensitivity (CSS_j) satisfies CSS_j< (MinimumSensRatio × CSSmax). Default=no.

MinimumSensRatio – Used as described for OmitInsensitive. Default=0.005.

ReincludeSensRatio – If ReincludeSensRatio>0.0 and CSS_j > (ReincludeSensRatio × CSSmax), reinclude parameter j. Default=0.02.

Keywords that control weights for observations with coefficients of variation for which simulated values can be used to calculate the weights.

TolParWtOS – Controls the parameter-change threshold for using observed or simulated values to calculate weights on observations by controlling calculation of A and B in Figure 3. Default=10.

TolParWtOSRenew - Determines what happens after the first time the maximum calculated parameter change is less than criterion B of Figure 3. Default=False.

Keywords that control the trust-region modification of Gauss-Newton regression.

TrustRegion - No, do not use the trustregion approach. Dogleg: use the double-dogleg modification. Hookstep: use the hookstep modification. Default=no.

MaxStep - Maximum allowable step size used in the trust-region method. The default is a function of the sensitivities and the parameter values, and is printed in the UCODE_2005 main output file.

ConsecMax - Maximum number of times that MaxStep is used consecutively before execution stops. Default=5.

Scaling - Yes, no. Yes, scaling is used when calculating the parameter change vector. Default = Yes

Reg_GN_NonLinInt Input Block (optional) UCODE_2005 Chapter 17

ConfidenceOrPrediction – confidence or prediction interval. Default=confidence.

IndividualOrSimultaneous – individual or simultaneous interval. Default=individual.

WhichLimits - Lower, Upper or Both. Default=Both.

TolIntP - Tolerance based on parameter values. Default=0.001.

TolIntS - Tolerance based on model fit. Default=0.1×TolIntP.

TolIntY - Tolerance based on change in the value of the computed interval limit. Default=0.001.

CorrectionFactors - yes, no. Yes: Use correction factors. Default=no.

The following keyword is used only if CorrectionFactors=yes.

AlternateStartValues – yes, no. Yes: Use starting parameter values from Parameter_Values input block. Default=no, which means use values in fn._paopt.

Model_Command_Lines Input Block (required) UCODE_2005 Chapter 6

Command - Operating system command that executes the process model(s).

Purpose - The type of process model run performed. Default=forward.

CommandID - A name for the command.

Parameter Definition

Parameter_Groups Input Block (optional) UCODE_2005 Chapter 7

GroupName - The name of the group (up to 12 characters; not case sensitive). Default=ParamDefault

Other keywords - Any keyword from the Parameter_Data input block.

Parameter_Groups_For_Prediction (optional) This report Chapter 2

The keywords are the same as for the Parameter_Groups input block.

Parameter_Data Input Block (required) UCODE_2005 Chapter 7

ParamName - Parameter name (up to 12 characters; not case sensitive)

GroupName - Group name (up to 12 characters; not case sensitive). Default=ParamDefault.

StartValue - Starting parameter value. Default=+(A huge real number).

LowerValue - Smallest reasonable value for this parameter. Default = -(A huge real number).

UpperValue - Largest reasonable value for this parameter. Default = +(A huge real number).

Constrain - yes, no. Yes: constrain the parameter value. Default=no.

UpperConstraint - Upper limit on the parameter value.

LowerConstraint - Lower limit on the parameter value.

Adjustable - yes, no. Yes: this parameter value can be changed for the purpose defined in the UCODE_CONTROL_DATA input block. Default=no.

PerturbAmt - Fractional amount of parameter value to perturb to calculate sensitivities for this parameter. Default=0.01.

Transform - yes, no. Yes: log-transform the parameter for the regression. Default=no.

TolPar **-** Replaces, for this parameter, the value of TolPar from the Reg_GN_Controls input block or the default of 0.01.

MaxChange - Maximum fractional parameter change allowed between parameter iterations. Default=2.0.

SenMethod - How sensitivities are obtained. -1=read as log transformed, 0=read as native, 1=forward perturbation, 2=central perturbation. Default=1.

ScalePval **-** A positive number used to scale sensitivities if the parameter value gets too small. Default=StartValue/100.

SOSIncrement **-** The number of values to be considered when SOSsurface=yes in the UCODE_CONTROL_DATA input block. Default=5.

NonLinearInterval – yes, no. Yes: calculate nonlinear intervals for this parameter when NonlinearIntervals=yes in the UCODE_CONTROL_DATA block. Default=no.

Parameter_Data_For_Prediction (optional) This report Chapter 2

The keywords are the same as for the Parameter_Data input block.

Parameter_Values Input Block (optional) UCODE_2005 Chapter 7

ParamName - The name of the parameter for which a value is specified.

StartValue - The specified parameter value.

Derived_Parameters Input Block (optional) UCODE_2005 Chapter 7

DerParName - Name of derived parameter (up to 12 characters; not case sensitive).

DerParEqn - An equation without an "equal" sign (that is, just the right-hand side of the equation) by which the derived parameter is calculated, generally using defined parameters.

Derived_Parameters_For_Prediction (optional) This report Chapter 2

The keywords are the same as for the Derived Parameters input block.

Observations (omit/ignored for prediction mode) UCODE_2005 Chapter 8

Observation_Groups Input Block (optional)

GroupName - Name for a group of observations (up to 12 characters; not case sensitive). Default=DefaultObs.

UseFlag - yes, no. Yes: use the simulated values in this group to compare against observed values in the regression. Default=yes.

PlotSymbol - An integer intended for use in post-processing programs to assign symbols for plotting. Default=1.

WtMultiplier - Value used to multiply weights for members of a group when the weights are defined using Statistic and StatFlag keywords of the Observation_Data input block. Default=1.0.

CovMatrix - Name of the error variance-covariance matrix.

Other keywords - Any keyword from the Observation_Data input block.

Observation_Data Input Block (required)

ObsName - Observation name (up to 20 characters; not case sensitive). Each observation name needs to start with a letter and to be unique.

ObsValue - Observation value.

Statistic - Statistic used to calculate the observation weight.

StatFlag - Defines Statistic. Options: VAR, SD, CV, WT, SQRWT. No default.

GroupName - Group name from the Observation_Groups input block. Default=DefaultObs.

Equation **-** An equation without an "equal" sign (just the right had side of the equation) that defines how to calculate an equivalent simulated value from simulated equivalents of previously defined observations. Default= _

NonDetect **-** Detection limit for an observation. Default=0.

WtOSConstant - The constant η in equation 1 of Chapter 3. Default=0.

WtOSUse **-** OBS, ConvertOS, SIM, or None. Default depends on WtOSConstant:
if WtOSConstant=0.0, WtOSUse=OBS;
if WtOSConstant>0.0, WtOSUse=ConvertOS.

Derived_Observations Input Block (optional)

The Derived_Observations input block is identical to the Observation_Data input block. It is included in UCODE so the user can define derived observations in a separate input block, which is convenient in some circumstances.

Predictions (required for selected modes) UCODE_2005 Chapter 8

Prediction_Groups Input Block (optional)

GroupName - Name for a group of predictions (up to 12 characters; not case sensitive). Default=DefaultPreds.

UseFlag - yes: report and analyze the predictions in this group. Default=yes.

PlotSymbol - An integer intended for use in post-processing programs to assign symbols for plotting. Default=1.

Other keywords - Any keyword from the Prediction_Data input block.

Prediction_Data Input Block (required for modes prediction, advanced-test-model-linearity, nonlinear-uncertainty modes)

PredName - Prediction name (up to 20 characters; not case sensitive).Each prediction name needs to start with a letter and to be unique.

RefValue - Reference value to which the prediction is compared.

MeasStatistic - A statistic used to calculate the variance of the measurement error.

MeasStatFlag - Defines MeasStatistic. Options: VAR, SD. No default.

GroupName - Group name from the Prediction_Groups input block.

Equation **-** An equation without an “equal” sign (just the right hand side) that defines how to calculate a derived prediction. Default= ‘_’.

Derived_Predictions Input Block (optional)

The Derived_Predictions input block is identical to the Prediction_Data input block except in name. It is included in UCODE_2005 so the user can define derived predictions in a separate block, which may be convenient in some circumstances.

Prior Information

Prior_Information_Groups Input Block (optional) UCODE_2005 Chapter 9

GroupName - Name for a group of prior information items (up to 12 letters, numbers, and _; not case sensitive). Default=DefaultPrior.

UseFlag - yes: include this group when estimating parameters. Default=yes.

PlotSymbol - An integer used in post-processing programs for the purpose of assigning symbols for plotting. Default=1.

WtMultiplier - Value that multiplies the weights for members of the group when the weighting is defined using Statistic and StatFlag keywords described for the Linear_Prior_Information input block. Default=1.0

CovMatrix - Name of the error variance-covariance matrix.

Other keywords - Any keyword from the Linear_Prior_Information input block.

Prior_Information_Groups_For_Prediction (optional) This report chapter 2

The keywords are the same as for the Prior_Information_Groups input block.

Linear_Prior_Information Input Block (optional) UCODE_2005 Chapter 9

PriorName - Prior information equation name (up to 20 letters, numbers, and _; not case sensitive; start with a letter). Default=DefaultPrior.

Equation - An equation without an "equal" sign that defines the prior information in terms of parameter names as specified in the Parameter_Data or Derived_Parameters input blocks.

PriorInfoValue - Value of prior information.

Statistic - Value used to calculate the prior information weight.

StatFlag - Defines Statistic. Options: VAR, SD, CV, WT, SQRWT. No default.

GroupName - Name for a group of prior information items.

WtOSUse **-** PRI, ConvertOS, or SIM. Default depends on WtOSConstant: if WtOSConstant=0.0, WtOSUse=PRI; if WtOSConstant>0.0, WtOSUse=ConvertOS.

WtOSConstant - The constant η in equation 3. Default=0.

Linear_Prior_Information_For_Predictions (optional) This report chapter 2

The keywords are the same as for the Linear_Prior_Information input block. Expressions defined for the Equation keyword are limited to using parameters defined in input blocks with names that end in "_For_Prediction"

Matrix_Files Input Block (optional) UCODE_2005 Chapter 10

MatrixFile - Name or path of the file from which one or more matrices are read. (Up to 2,000 characters; case sensitivity depends on the operating system).

NMatrices - Number of matrices to read from MatrixFile**.** Default=1.

Complete Matrix

```
CompleteMatrix  [NAME]
NGMEM  NGMEM  [ControlRecord]
[Array Control Record]
VAL(1,1)  . . . . . . . VAL(1,NGMEM)
   .     .                .
   .          .           .
   .               .      .
VAL(NGMEM,1) . . . . . . . VAL(NGMEM,NGMEM)
```

Compressed Matrix

```
CompressedMatrix  [NAME]
NNZ  NGMEM  NGMEM [ControlRecord]
[Array Control Record]
IPOS(1)    VAL(1)
IPOS(2)    VAL(2)
...
IPOS(NNZ)  VAL(NNZ)
```

Array Control Record Input Instructions

1. ***INTERNAL*** CNSTNT FMTIN IPRN

2. ***OPEN/CLOSE*** FNAME CNSTNT FMTIN IPRN

Model_Input_Files Input Block (required) UCODE_2005 Chapter 11

ModInFile - Name for a process-model input file (up to 2,000 characters). Names with spaces need to be enclosed in double quotes.

TemplateFile - Name for the template file (Up to 2,000 characters. Case sensitivity depends on the operating system). Names with spaces need to be enclosed in double quotes.

Template Files (required) UCODE_2005 Chapter 11

A template file is created from a model input file by first inserting a line at the top. The line contains "jtf" followed by one or more spaces and the substitution delimiter. Commonly used substitution delimiters are @ and !.

The substitution delimiter is used to define the space within which UCODE_2005 places a number. The substitution space is defined by a pair of substitution delimiters. All of the characters between and including the substitution delimiters are replaced. Characters between the delimiters need to include spaces and one ParamName or DerParName defined in the Parameter_Data or Derived_Parameter input block. The ParamName or DerParName can be placed anywhere between the delimiters.

Model_Output_Files Input Block (required) UCODE_2005 Chapter 11

ModOutFile - Name of the process-model output file with values to be extracted. Up to 2,000 characters; case sensitivity depends on the operating system.

InstructionFile - Name for the Instruction file that UCODE_2005 uses to extract values from ModOutFile. InstructionFile can be up to 2,000 characters; case sensitivity depends on the operating system.

Category - Identifies the type of quantity for which values are extracted. **Obs:** observations. **Pred:** predictions

---

Instruction Files (required) UCODE_2005 Chapter 11

For a Standard Process-Model Output File

```
jif @
StandardFile Nskip ReadColumn Nread
[Names for each of the Nread values.
Place each name on a new line.]
```

Nskip, number of lines to skip at top of file.
ReadColumn, the column to be read.
Nread, the number of items to be read.

For a Non-Standard Process-Model Output File

The first line of an instruction file needs to begin with the three letters “jif”, a single space, and a marker delimiter. Usually $, @, % or ~ are good choices for marker delimiter.

Except for ‘dum’, which can be used repeatedly, a different name needs to be used for each extracted value.

A complete list of instructions is presented in Table 9, Chapter 11 of Poeter et al. (2005).

Parallel Processing (optional) UCODE_2005 Chapter 12

Parallel_Control Input Block

Parallel	- yes, no. Yes: Activates parallel processing. Default=no.
Wait	- Time delay, in seconds, used in file management. Default=0.001.
VerboseRunner	- Flag that controls printing by the runner. Default=3.
AutoStopRunners	-yes, no. Yes: stop runners when UCODE_2005 stops. Default=yes.
OperatingSystem	- Operating system for dispatcher and runner. Default=Windows.
TimeoutFactor	- Factor for RUNTIME to identify overdue run. Default=3.0.

Parallel_Runners Input Block

RunnerName	- Name of runner. Up to 20 characters.
RunnerDir	- Pathname to directory where the runner program runs.
RunTime	- Expected model runtime, in seconds. Default=10.

Equation Protocols (optional) UCODE_2005 Chapter 13

In UCODE_2005, equations can be defined in the Derived_Parameters, Observation_Data, Derived_Observations, Prediction_Data, and Derived_Predictions input blocks. See UCODE-2005 Chapter 13 for additional information.

Derivatives Interface Input File (optional) UCODE_2005 Chapter 13

A Derivatives Interface input file provides UCODE_2005 with information needed to obtain model-calculated sensitivities (derivatives of simulated values with respect to parameters) from a model-output file rather than determining them by perturbation. See Chapter 13.

fn.xyzt Input File (optional) UCODE_2005 Chapter 13

The first line of the xyzt input file is ignored. The rest of the file needs to be composed of lines containing five columns of data: Observation name, x, y, z, and time.

Reference

Poeter, E.P., Hill, M.C., and Banta, E.B., 2005, UCODE_2005 and six other computer codes for universal sensitivity analysis, calibration, and uncertainty evaluation: U.S. Geological Survey Techniques and Methods, book 6, sec. A, chap. 11, 283 p. http://igwmc.mines.edu/freeware/ucode/