Title
Solving iTOUGH2 simulation and optimization problems using the PEST protocol

Author
Finsterle, S.A.

Publication Date
2011-03-20

Peer reviewed

Solving iTOUGH2 Simulation and Optimization Problems Using the PEST Protocol

Stefan Finsterle* and Yingqi Zhang

Lawrence Berkeley National Laboratory, Earth Sciences Division,

1 Cyclotron Road, MS 90-1116, Berkeley, CA 94720, USA

Abstract

The PEST protocol has been implemented into the iTOUGH2 code, allowing the user to link any simulation program (with ASCII-based inputs and outputs) to iTOUGH2's sensitivity analysis, inverse modeling, and uncertainty quantification capabilities. These application models can be pre- or postprocessors of the TOUGH2 non-isothermal multiphase flow and transport simulator, or programs that are unrelated to the TOUGH suite of codes. PEST-style template and instruction files are used, respectively, to pass input parameters updated by the iTOUGH2 optimization routines to the model, and to retrieve the model-calculated values that correspond to observable variables. We summarize the iTOUGH2 capabilities and demonstrate the flexibility added by the PEST protocol for the solution of a variety of simulation-optimization problems. In particular, the combination of loosely coupled and tightly integrated simulation and optimization routines provides both the flexibility and control needed to solve challenging inversion problems for the analysis of multiphase subsurface flow and transport systems.

Keywords: Optimization; sensitivity analysis; inverse modeling; uncertainty quantification; iTOUGH2; PEST

* Tel.: +510-486-5205; fax: +510-486-5686; e-mail address: SAFinsterle@lbl.gov

1. Introduction

Simulation models are essential tools in environmental science and engineering. They are used for scientific hypothesis testing, design of laboratory and field experiments, site characterization and data analysis, hind- and forecasting, risk assessment and decision support. Models in general and environmental models in particular are abstracted representations of a complex system, where certain aspects—properties, features, processes, controls—are represented by approximate equations and (model-related) effective parameters. Parameterization is a key part of conceptual model development. In addition to the accuracy of the conceptual model, the ability of a model to reproduce historical data or to adequately predict future system behavior critically depends on (1) the number of parameters, (2) the consistency between the model parameter and the aspect of the real system the parameter is supposed to represent, (3) the parameter's actual value and the way it was determined, and (4) its relation to other (adjustable and fixed) parameters. Doherty and Welter (2010) provide an excellent discussion of these and other parameterization issues.

Simulations are often performed with one or more of its input parameters changed in a random or systematic manner to (1) evaluate the parameter's impact on model output (sensitivity analysis), (2) determine their value based on measured data (parameter estimation, history matching, inverse modeling), (3) examine design alternatives or to optimize operational activities (optimal design), or (4) quantify accuracy and reliability of model predictions (uncertainty quantification). The following elements are common to these analyses: (1) Parameters need to be selected or defined; they may be identical to the primary parameters used in the model, or comprised of multiple, potentially transformed primary parameters; (2) output variables need to be selected or defined; they may be directly calculated by the model, or be an aggregate of multiple, potentially transformed primary output variables; (3) one or multiple models are needed to relate the primary input parameters to the primary output variables; and (4) an algorithm is needed to generate or update the parameter values based on input information, the predicted output, or other rules and criteria.

The iTOUGH2 code (http://www-esd.lbl.gov/iTOUGH2) provides inverse modeling capabilities for the non-isothermal, multiphase, multicomponent flow and transport simulator TOUGH2 (Pruess et al., 1999; Finsterle et al., 2008). iTOUGH2 has been extensively used for the analysis of synthetic, laboratory, and field data for applications related to geothermal reservoir engineering (Kiryukhin et al., 2008), nuclear waste isolation (Ghezzehei et al., 2004), geologic carbon sequestration (Zhang et al., 2011), environmental remediation (Linde et al., 2006), fractured rock hydrology (Finsterle et al., 2002; Unger et al., 2004), landfill management (Jung et al., 2011), vadose zone hydrology (Kowalsky et al., 2005), geotechnical engineering (Moridis et al., 1999; Gallagher and Finsterle, 2004), water resources management (Zhang et al., 2010) and other application areas (for a review, see Finsterle (2004)).

While the original iTOUGH2 code is tightly linked to the TOUGH2 simulator, its optimization routines are general enough to be coupled to any forward model. This concept has long been followed by general, model-independent, nonlinear parameter estimation packages such as PEST (Doherty, 2008; http://pesthomepage.org/) and UCODE (Poeter and Hill, 1998; http://water.usgs.gov/software/ucode.html). Both of these widely used universal codes are based on the PEST protocol (Doherty, 2008; Banta et al., 2008), which defines the interface between the analysis tool and the input and output files of the application software. To make iTOUGH2 capabilities accessible to more application models, the subroutines comprising the PEST protocol—provided by Doherty (2007; http://www.pesthomepage.org/getfiles.php?file=modules.zip)—have been implemented into iTOUGH2.

The concept behind the PEST protocol requires the application model (1) to provide input through one (or more) ASCII input files, (2) to return output to one (or more) ASCII output files, (3) to run the model (or multiple models) using a system command (an executable or script/batch file), and (4) to run the models to completion without any user intervention. For each forward run invoked by iTOUGH2, selected parameters in the application model input files are overwritten with values updated by iTOUGH2, and selected variables in the output files are extracted and returned to iTOUGH2. The core of iTOUGH2, i.e., its optimization routines and related analysis tools, remains unchanged; only the communication format between input parameters, the application model, and output variables are borrowed from PEST. The inclusion of the PEST protocol into the iTOUGH2 architecture is shown in Figure 1. The parameter vector (which is updated by the minimization algorithm or by the sampling procedure used for uncertainty quantification) is transferred to the PEST protocol, which replaces generic parameter names in the so-called template file with the appropriate numerical values and generates a valid input file. The external model is executed using a system call, which may be a command line, the name of an executable code, or a script file. After completion of the model run, the resulting output files are parsed using directives from the PEST instruction file, and the values of interest are extracted and filled into the observation vector, which is then used by iTOUGH2 to evaluate the objective function or for further analysis. The extended code allows the user to invoke optimization of TOUGH2 models, which are fully integrated within iTOUGH2, or any external models, which are loosely linked by the PEST protocol, or a combination thereof.

We first summarize the iTOUGH2 optimization and analysis capabilities, which are now also available in combination with any simulation code that uses ASCII input and output files (or keyboard input and text output to the screen). We then discuss some examples that demonstrate the use of the PEST interface. These illustrative test cases make use of external multiphase simulators from the TOUGH2 suite of code; the extension to other simulation software is straightforward.

2. iTOUGH2 Capabilities

iTOUGH2 was originally designed to provide inverse modeling capabilities for the TOUGH2 suite of non-isothermal multiphase flow simulators (Finsterle, 2004; Finsterle et al., 2008). Sensitivity coefficients calculated as part of the gradient-based or second-order minimization algorithms can also be used to examine the information content of actual or planned observations, to evaluate the design of an experiment or monitoring network, and to study the impact of parameter uncertainty on model predictions. These and most other iTOUGH2 capabilities described in this section are also applicable to any model that can be linked to iTOUGH2 through the PEST protocol; a few features are specific to the parameterization and prediction variables of the TOUGH2 codes. For example, lists of TOUGH2 elements, connections, sinks and sources, and material types can conveniently be grouped and defined as a single parameter to be estimated or a single integrated observation to be matched (iTOUGH2 also allows the user to combine disparate parameters).

2.1 Parameterization

In the context of this paper, parameters are defined as adjustable variables that represent those aspects of a model that are subjected to sensitivity analysis, parameter estimation, or uncertainty propagation analysis. These parameters may refer to material properties, initial and boundary conditions, or geometric features (such as fracture spacing, or the location and shape of discrete zones). Heterogeneity may be parameterized using a relatively small number of geostatistical parameters (Finsterle and Kowalsky, 2008). Moreover, statistical properties (e.g., autocorrelation coefficients, Box-Cox parameters), weighting coefficients, and correction terms may also be considered parameters to be estimated (Finsterle and Zhang, 2011). Parameters may directly correspond to an input variable of the model, or represent a collection of properties and features, i.e., a single estimated parameter may be linked to multiple input variables. Parameters may also be transformed (e.g., by taking the logarithm, or by estimating a factor with which multiple input variables are multiplied). It is important to realize that any other model input that is fixed during an inversion becomes part of the model structure. The values and uncertainties of the parameters to be estimated always refer to this specific model structure. While parsimonious models with few parameters are often used to avoid overparameterization, their ability to make predictions is limited to models with the same or very similar model structure, as structural errors in the calibration model are partly absorbed by the estimated parameters. This makes these parameters tailored to that specific model and thus less suitable for predictive calculations with a changed model structure.

In iTOUGH2, most input values to the TOUGH2 simulators can be accessed directly through built-in commands. Moreover, an application programming interface (API) is provided to define user-specified parameters. All these parameters are internally transferred between the simulation and optimization routines without loss of precision. With the PEST interface, any input variable can be accessed (with a potential loss of precision) through ASCII files, which are written by means of so-called template files. All these parameters can be tied to each other, and some basic transformations can be performed (add, multiply, logarithm, and combinations thereof). For each parameter, the user can specify a prior value and associated standard deviation (for regularization), an initial guess (for starting the optimization), lower and upper bounds (for specifying the admissible parameter range), an expected variation (for sensitivity analyses), and a probability distribution (for uncertainty quantification). In summary, essentially any input parameter to any TOUGH2-related or numerical model with ASCII input and output files can be subjected to iTOUGH2 analyses.

2.2 Observable Variables

All main iTOUGH2 application modes (i.e., sensitivity analysis (SA), parameter estimation (PE), and uncertainty quantification (UQ)) examine the response of specific model output variables to variations in selected input parameters. Specifically for PE, the model output of interest is a generally small subset of the simulation results: It only consists of those output variables for which a corresponding measured data point is available. We therefore refer to this set of model output as the observable variables. Observable variables may refer to the calculated system state at a specific point in space and time, or is a transformed, integral measure derived from several model output variables (for example, costs). A special type of parameter-dependent output variables of interest are regularization and penalty terms.

In iTOUGH2, most output variables from a TOUGH2 simulation can be accessed directly, i.e., from memory without loss of precision, through built-in commands. In addition, interfaces to geophysical forward models and their associated data types are implemented to perform joint hydrogeophysical inversion (Kowalsky et al., 2004; 2005; 2011; Finsterle and Kowalsky, 2008). Moreover, an API is provided to define user-specified observations. With the PEST interface, any output variable written to one or multiple ASCII files can be accessed by means of so-called instruction files. The potential loss of precision during this transfer (due to a limited number of significant digits printed to the output file) may be a critical shortcoming. Basic transformations of the observable variables can be performed, such as add, multiply, Box-Cox transformation (see Finsterle and Zhang, 2011), and combinations thereof. For each observational variable or entire data set, the user can specify a weight to be applied during the optimization procedure.

In summary, essentially any output variable of any TOUGH2-releated or text-based numerical model can be analyzed by iTOUGH2 or used for parameter estimation by automatic model calibration. As in any inverse problem, the latter only yields meaningful results if the observable variables are sufficiently sensitive and linearly independent with respect to the parameters of interest. The iTOUGH2 features discussed in the following subsections support such analyses.

2.3 Sensitivity Analysis

A sensitivity analysis (SA) consists of examining the impact of the model output variables with respect to changes in model input parameters. Such an analysis is useful to identify the key parameters of the system, to detect observations that may be suitable for parameter estimation, and to recognize which output is most strongly affected by uncertainties in input parameters. Moreover, sensitivity coefficients are used by derivative-based minimization algorithms to obtain the search direction along which parameters are updated to approach the optimum solution. This latter use prompts the calculation of an $m \times n$ Jacobian matrix $\mathbf{J}$ in iTOUGH2, with elements $J_{ij} = \frac{\partial z_i}{\partial p_j}$. Here, n is the number of input parameters, p, and m is the number of observable variables, z. To make sensitivity coefficients dimensionless and thus comparable with one another, they are scaled by the expected parameter variation, σ_p, and the inverse of the *a priori* standard deviation of the observation, σ_z $\tilde{J}_{ij} = J_{ij} \frac{\sigma_{p_j}}{\sigma_{z_i}}$. The columns of the Jacobian are calculated (in parallel) by forward or centered finite differences. Once the scaled sensitivity coefficients are available, integral measures of overall parameter sensitivity, or overall information content of individual observations, data sets, or observation types can be calculated. The Jacobian matrix is also used to compute the Fisher Information matrix ($\mathbf{F} = \mathbf{J}^T \mathbf{Q}_{zz}^{-1} \mathbf{J}$, where $\mathbf{Q}_{zz}^{-1}$ is the observation weighting matrix), which in turn reveals expected estimation uncertainties, the correlation structure among parameters, and parameter identifiability as defined by Doherty and Hunt (2009). Finally, the relative significance of each observation point to the solution of the inverse problem—using as the criterion the D-optimality of the estimation covariance matrix—is evaluated. All these measures can be used in support of experimental design prior to data collection, or to examine the quality of an inversion.

It is important to realize that all the sensitivity measures calculated by iTOUGH2 are based on local sensitivity coefficients as well as linearity and normality assumptions. For highly nonlinear systems or large parameter variations, methods and sampling designs that more fully explore the parameter space need to be used, so that the sensitivity measures are more robust and representative.

2.4 Objective Function

The objective function is a measure of misfit between the model results and the measured data. Prior information, regularization, and penalty terms may also be added. If assumptions about the stochastic structure of the residuals can be made, minimizing the appropriate objective functions leads to maximum likelihood estimates. While seldom explicitly stated or its appropriateness demonstrated, it is common to make a normality assumption and thus use the weighted least squares criterion as the performance measure to be minimized. Despite its popularity, an estimate based on least squares has the drawback of being potentially affected by violations of the underlying distributional assumptions. In particular, the presence of outliers in the data may lead to poor matches of the “good” data, which induces a bias of the estimated model parameters. Given the fact that field measurements show many more outlier points than one would expect from the tail of the normal distribution, their potential impact on inverse modeling results should be carefully assessed. In addition to the standard weighted least-squares objective function, iTOUGH2 offers robust estimators, such as the least absolute value, Andrews, Huber's and Cauchy estimators (Finsterle and Najita, 1998; Finsterle and Zhang, 2011). The L_1-estimator is also the preferred option when using iTOUGH2 for the solution of cost optimization problems (Finsterle, 2005).

The residuals need to be weighted, where the weights are often related to the distributional assumptions about the errors. They can be specified for individual residuals, entire data sets, or given as a function of the measured value. They can also be dynamically adjusted according to the procedure described in Carrera and Neuman (1986).

2.5 Minimization Algorithms

iTOUGH2 solves the inverse problem by finding the minimum of the objective function in the n-dimensional parameter space. The minimization algorithms currently implemented in iTOUGH2 are summarized in Table 1. They include derivative-based local algorithms as well as metaheuristic, derivative-free global search methods. For computationally expensive forward models, global optimization is often impractical, and the high efficiency of the derivative-based methods, specifically that of the Levenberg-Marquardt algorithm, is needed to identify the local minimum of a carefully formulated inverse problem. If many, potentially strongly correlated parameters are subjected to the estimation process, a dynamic parameter selection and conditioning scheme is implemented based on the parameter's relative sensitivity and overall correlation. A composite scaled sensitivity measure (i.e., the sum of the absolute values of all weighted sensitivity coefficients) is calculated for each parameter. Similarly, a measure of overall parameter correlation (i.e., the ration of the conditional to marginal estimation standard deviation) is evaluated. All parameters with a sensitivity or correlation measure less than a certain fraction of the most sensitive or least correlated parameter are temporarily moved to the parameter null space. An alternative approach to dynamically delineate the parameter solution from the parameter null space is described in Finsterle and Kowalsky (2011).

2.6 Residual and Error Analysis

Even if the minimization algorithms described above successfully identified the local or global minimum of the objective function, this does not guarantee that (1) the match to the data is satisfactory and the model is a good representation of the actual system, (2) the estimated parameters values are reasonable, and (3) the estimation and prediction uncertainties are acceptable. A detailed residual, error, and uncertainty analysis is needed to assess the inverse modeling results, and to gain insights into the system behavior and its dependence on parameters, which can point towards aspects of the model that may need to be refined. Some of the methods used to analyze residuals after an iTOUGH2 optimization are described in Finsterle and Zhang (2011). The covariance matrix of the estimated parameters is calculated based on a linearity and normality assumption. A correction procedure to account for nonlinearities□originally proposed by Carrera (1984)□is also implemented. Prediction uncertainty is evaluated using linear uncertainty propagation analysis or Monte Carlo simulations using a Latin Hypercube sampling strategy that allows the inclusion of parameter correlations (Zhang and Pinder, 2003; Kitterød and Finsterle, 2004).

2.7 Relation between iTOUGH2 and PEST

The inclusion of the PEST protocol in iTOUGH2 does not imply that any of the PEST optimization capabilities are implemented in iTOUGH2; the sole purpose of the PEST protocol is to make iTOUGH2 optimization routines available for use in connection with external forward models. In general, parameter estimation codes such as PEST, UCODE, and iTOUGH2 all aim at solving highly nonlinear least-squares problems for computationally expensive forward models. Consequently, the inverse modeling capabilities of these codes are similar; the significance of the differences among these codes depends on the needs of a specific application. Both PEST and iTOUGH2 contain versions of the Levenberg-Marquardt algorithm with the ability to truncate the parameter space; the method used to reduce the impact of parameters with strong correlations or low sensitivities, however, are different. The concept of estimating superparameters (Tonkin and Doherty, 2005), implemented in PEST, is a powerful method to address highly parameterized inverse problems. The regularization approach employed by iTOUGH2 is described in Finsterle and Kowalsky (2011). In addition to the Levenberg-Marquardt algorithm, iTOUGH2 provides the local and global minimization methods summarized in Table 1. Both PEST and iTOUGH2 provide geostatistical methods to parameterize heterogeneity, and the pilot-point approach to adjust these property fields to match the observed system response. Both codes perform a rather extensive residual and uncertainty analysis as a basis to evaluate prediction errors. Parallel execution of independent forward simulations is supported by both software packages. Details of the implementation of these capabilities as well as the amount of user control and convenience of input are specific to each of these codes. Because PEST, iTOUGH2, and other similar packages are continually updated, the user is referred to the respective user's guides for detailed capability descriptions.

The use of the PEST protocol to estimate parameters of iTOUGH2 pre- and postprocessing software in combination with the estimation of standard TOUGH2 parameters (an example is shown in Section 3.2 below) is a unique capability; it combines the loosely coupled and tightly integrated approaches to parameter estimation, and greatly expands the flexibility to calibrate and analyze TOUGH2 models.

Both PEST and iTOUGH2 are mainly concerned with inverse problems where the evaluation of the forward model is computationally very expensive. This essentially precludes the use of stochastic, sampling-based parameter estimation approaches, even though the potential of such approaches to evaluate posterior probability density functions addresses an important parameter estimation issue. We are currently working on the implementation of statistical sampling approaches to perform global sensitivity analyses, advanced uncertainty quantification, and global optimization within a Bayesian framework; we will report on these advances in due course.

3. iTOUGH2 Applications using PEST Protocol

The following examples demonstrate potential usages of the PEST protocol in combination with iTOUGH2 analysis and optimization routines. The discussion focuses on the code's features rather than on the scientific contents of the individual analyses. The three examples make use of, respectively, iTOUGH2's parameter estimation, uncertainty quantification, and grid search capabilities. The external codes used are TOUGHREACT, a script file invoking various mesh generation steps as a preprocessor to a TOUGH2 simulation, and an iTOUGH2 inversion itself. The PEST template files modify the TOUGHREACT input file that holds chemical properties, and input file with statistical parameters for the generation of a discrete fracture network, and the weighting coefficients in an iTOUGH2 input file. Although the examples consider simulations of flow and transport in the subsurface, iTOUGH2 with the PEST protocol can be used to solve optimization problems for any type of simulation application.

3.1 Multicomponent Reactive Transport Inversion and Comparison with PEST

iTOUGH2 provides inverse modeling capabilities for many but not all of the members of the TOUGH family of non-isothermal multiphase flow simulators. A list of publicly available modules that are fully integrated into iTOUGH2 can be found on the TOUGH+ web site at http://esd.lbl.gov/TOUGH+/software-itough2.html. The PEST protocol expands inverse modeling to all TOUGH-related simulators, specifically TOUGHREACT.

This example demonstrates parameter estimation using the TOUGHREACT simulator (Xu et al., 2004) and the parallel version iTOUGH2-PVM (Finsterle, 1998). TOUGHREACT is a model for the simulation of nonisothermal multiphase flow and reactive transport in fractured porous media.

In this example, TOUGHREACT is applied to simulate urea hydrolysis (ureolysis) as a means to remediate ^{90}Sr contamination in the saturated zone (e.g., Fujita et al., 2000, 2004; Mitchell and Ferris, 2005). This simulation involves the modeling of a ureolysis column experiment (Wu et al., 2010) in which water with added urea was injected into a soil column (for about 15 days), while the water composition at the column outlet was monitored and compared to model results. Ureolysis consumes hydrogen ions and produces ammonium and bicarbonate ions. Consequently, the injection of urea into the column causes pH and alkalinity to increase, driving calcite precipitation. Strontium, which strongly partitions into soils, exchanges with ammonium ions produced by ureolysis and precipitates with calcite. These coupled biogeochemical processes are modeled with TOUGHREACT. The reaction network and model input parameters are described in Spycher et al. (2009) and Wu et al. (2010). The data were originally inverted using Parallel Pest (PPEST; Doherty, 2008). These results are used for comparison with inversions of the same simulation using iTOUGH2. The one-dimensional column is discretized into 205 gridblocks at regularly spaced intervals of 1 mm. A sequential-iterative (transport/reaction) method is implemented. The model considers ureolysis as an enzymatic reaction and accounts for calcite precipitation, ion exchange, and ammonium oxidation. Further details about the system behavior and the TOUGHREACT model can be found in Spycher et al. (2009) and Wu et al. (2010).

The following five parameters (see Table 2) are estimated by inverse modeling: (1) the initial and boundary concentration of the urease enzyme, which directly affect the ureolysis rate, (2) the initial and boundary concentration of the biomass, which affects the oxidation rate of produced ammonium ions, (3) the logarithm of the precipitation rate constant for calcite, (4) the exchange coefficient (selectivity) of potassium, and (5) the soil cation exchange capacity. These five parameters are input into TOUGHREACT through the ASCII input file *chemical.inp*, which holds all geochemical parameters and properties of the aqueous component species, minerals, gases, and sorbed species for a given simulation. On running iTOUGH2, this file (*chemical.inp*) is automatically generated by a PEST template file. The template file takes the same format as the regular input file, except that the values of the five parameters to be estimated are replaced with the corresponding variable names, surrounded by a special character chosen as the parameter delimiter.

These five parameters are then estimated by matching breakthrough curves of measured concentrations of pH, NH_4^+, NO_3^-, dissolved O_2, Urea, Ca, Sr, Na, and K. A PEST instruction file is used to instruct iTOUGH2 on the location of the calculated pH and concentrations in the TOUGHREACT output file. iTOUGH2 can then read the TOUGHREACT output after each successive forward run to compare computed and observed pH and concentration values.

The standard iTOUGH2 control file is used to relate the template and instruction files to the appropriate TOUGHREACT input and output files, respectively. Moreover, the parameters to be estimated as well as the observed data are defined using the standard iTOUGH2 commands (see the command index at http://esd.lbl.gov/iTOUGH2). Finally, inversion options are selected and computational parameters provided. In this case, five Levenberg-Marquardt iterations are performed, where the columns of the Jacobian matrix and the evaluation of a potential update step with different Levenberg parameters λ are performed in parallel using PVM (Finsterle, 1998).

The inversion results are summarized in Table 2 and compared to the results obtained with Parallel PEST (PPEST; Doherty, 2008). Both codes converged to the same objective function value and the same solution in the parameter space. The differences between the estimated parameters are a result of the different implementation of the Levenberg-Marquardt algorithm in iTOUGH2 and PPEST, and specifically the different default values of computational parameters (such as the initial values of the Levenberg and Marquardt parameters, step size limitations, etc.). However, these differences are much smaller than the estimation uncertainty, which is also consistently calculated by the two optimization codes. With PPEST, almost twice as many TOUGHREACT forward runs were required as with iTOUGH2, mainly because PPEST switched to central finite differences for evaluating derivatives after two iterations, which also explains the (small) differences in the calculated estimation uncertainty.

This particular inversion took approximately 16 hours to complete on a Linux cluster. Almost all the CPU time is consumed by repeatedly running the TOUGHREACT simulation model; only a negligible CPU fraction is used by the minimization algorithm, residual, and uncertainty analyses. Evaluating the Jacobian matrix and testing Levenberg parameters in parallel on five processors sped up the inversion by a factor of 2.5. In this case, the parallelization yields a moderate gain in overall performance because of the relatively small number of parameters to be estimated.

This example demonstrates that parameters of a complex reactive transport simulator can be estimated using iTOUGH2, and that the results are consistent with the PEST estimates.

3.2 Analyzing Seepage using Multiple Discrete Fracture Network Models

iTOUGH2 can be used to simultaneously adjust parameters of an external model and an internal TOUGH2 model. This is useful if the external model is either a pre- or post-processor of TOUGH2. In this example we combine a pre-processor for generating realizations of a discrete fracture network with a TOUGH2 simulation of water seeping into an underground opening. The parameters to be considered uncertain and adjusted by iTOUGH2 are the stochastic parameters used by the mesh generator, i.e., the fracture density and parameters of the probability distributions from which length and orientation of two fracture sets are sampled. Selected output from both the external mesh generator (here, the number of fractures) and the flow simulator (seepage into the opening excavated from the fractured formation) are evaluated for an uncertainty analysis. Multiple steps are needed to generate a discrete fracture network model (see Table 3). These mesh generation steps are executed by a Linux shell script file *sh.DFNMgen*; it is the executable called by iTOUGH2 prior to each TOUGH2 forward simulation.

The fracture network consists of two fracture sets generated using six statistical parameters: the fracture trace length follows a power-law distribution (Bonnet et al., 2001), with the coefficient α and exponent $-a$ as its parameters; the orientations of the two fracture sets follow normal distributions, each with a given mean and standard deviation. Fracture aperture—and thus permeability—is correlated to the fracture length (for details, see Liu et al., 2002), with increasing permeabilities in the excavation disturbed zone as a linear function of distance from the opening.

Once the base fracture network has been generated, unconnected fractures are removed, the fracture traces are discretized according to the TOUGH2 spatial discretization scheme, an opening representing an excavated niche is cut form the mesh, and boundary elements are created. The output from these mesh generation steps is a file *MESH* that is read by TOUGH2 for the subsequent simulation of unsaturated flow through the discrete fracture network and seepage into the niche. Figure 2 visualizes the sequence of mesh generation steps, and shows some realizations obtained by varying the statistical input parameters. The permeability and steady-state saturation fields are also shown. The main output of interest is the steady-state seepage rate into the niche, which is obtained directly from the corresponding TOUGH2 variable using standard iTOUGH2 commands. In addition, the total number of fractures of the base network and the number of connected fracture are extracted from the output files of the mesh generator using an appropriate PEST instruction file.

Figure 3 shows the section of the iTOUGH2 input file in which program options and computation parameters are specified. In this application, the execution of 500 Monte Carlo simulations based on the Latin hypercube sampling strategy is used to examine the impact of the characteristics of the discrete fracture network on seepage. A covariance/correlation matrix of the six PEST parameters is provided, with the variances on the diagonal, and correlation coefficients on off-diagonal elements. Here, it is assumed that the two fracture sets are approximately orthogonal to each other; a correlation coefficient of 0.9 between the third and fifth parameters (those representing the mean angles for each fracture set) induces this statistical correlation. A weaker correlation coefficient of 0.5 is given for the respective standard deviations. The executable to run is specified in the iTOUGH2 input file (Figure 3). In the present example, the executable is the script file *sh.MESHgen*; it will be run before each TOUGH2 simulation. Other entries in the iTOUGH2 input file include the names of the PEST template and instruction files and their corresponding input and output files, as well as run specifications. Here, 500 Monte Carlo simulations are evaluated in parallel on 30 processors on a Linux cluster. The names of the nodes are stored on file *NODEFILE*, which is generated by the scheduler.

Figure 4 shows the results of the analysis. The histogram in Figure 4a shows that the total number of fractures (and the number of connected fractures) varies from about 150 to 300 as a result of uncertainty in the stochastic input parameters used to generate the fracture network. The changes in the characteristics of the fracture network impact the amount of water seeping into the underground opening (Figure 4b). This impact, however, is relatively mild. This is a result of the fact that the primary factor affecting seepage is the overall size and geometry of the opening, which is not uncertain. Changes in the uncertain statistical parameters have to lead to substantially changed network characteristics to be able to affect seepage. This explains why the seepage distribution is relatively peaked, and why a stochastic continuum representation is appropriate for seepage predictions (Finsterle, 2000). A detailed discussion of issues related to the modeling of seepage into a large opening from an unsaturated fractured formation can be found in Wang et al. (1999), Liu et al. (2002), Finsterle et al. (2003), Ghezzehei et al. (2004).

3.3 Pareto Frontier

The Pareto frontier can be considered to be the set of solutions to a multicriteria optimization problem, where the relative weights of the criteria are varied to examine the tradeoffs between competing objectives. Here, we determine the Pareto frontier by running multiple iTOUGH2 inversions, where the relative weights are adjusted in predefined increments. The grid-search option of iTOUGH2 is used, where the parameter to be varied is the weight assigned to two observation types, each representing a different objective. For each weight combination, an iTOUGH2 inversion is performed, and the mean residual of each observation type is extracted and used to create the Pareto frontier plot. In this example, iTOUGH2 controls iTOUGH2 optimization runs through the PEST protocol.

The optimization problem considered is a remediation design problem, where the tradeoff between two objectives is examined. These competing objectives are (1) maximization of contaminant removal within a specified cleanup time of 5 years, and (2) minimization of cleanup costs, simplified here as the total amount of water pumped from six wells during a pump-and-treat operation. The individual minimization problem of determining optimal pumping rates (assuming that the relative costs of pumping and residual contamination are known) is described in Finsterle (2005). This optimization problem is then solved repeatedly for different weights of the two competing objectives. By giving higher weight to the remediation goal, pumping rates are expected to go up; conversely, if emphasis is placed on reducing pumping costs, the pumping rates will generally go down at the expense of increased residual contamination. The tradeoff between these two objectives is evaluated at 40 discrete points with relative weights (w_p and w_c) for the pumping cost and remediation objectives, respectively, under the constraint that $w_p + w_c = 1$. The only parameter adjusted is the weight of the pumping rate criterion, w_p; its value is varied from zero to one. The second parameter (representing the weight given to the residual contamination criterion) is not a free parameter. It is tied to the first parameter using the equation $w_c = 1 - w_p$. The weights are entered into the iTOUGH2 input file, which is created by the PEST template file. For each weight combination, the optimal distribution of pumping rates in the six wells is determined by an iTOUGH2 optimization that minimizes both the (weighted) total amount of water pumped and the (weighted) residual contaminant mass. The total rate and residual contaminant mass after each optimization is extracted from the residual analysis section of the iTOUGH2 output file using a PEST instruction file. Plotting the two objectives against each other provides the Pareto frontier.

The 40 iTOUGH2 inversions are invoked through the standard Unix script command *itough2* (or the equivalent WINDOWS batch file), which is provided as the executable.

The resulting Pareto frontier is shown in Figure 5, demonstrating that there is a relatively well-defined optimal solution (i.e., the region of the Pareto frontier near the origin), where both criteria can be met without too much tradeoff.

4. Concluding Remarks

In the indirect approach to inverse modeling, optimization algorithms are wrapped around the numerical model whose parameters are to be estimated based on select output variables calculated by this model. Similarly, sensitivity analyses and uncertainty propagation analyses (specifically sampling-based methods) often treat the underlying forward operator as a black-box model. The fact that the optimization algorithms generally can be decoupled from the algorithms that solve the forward problem provides great flexibility in applying them to a large variety of scientific analysis and engineering design problems.

The applicability of the iTOUGH2 simulation-optimization code has been expanded by allowing the user to link it to any stand-alone modeling software with ASCII-based input and output by means of the widely-used PEST protocol. Using the PEST protocol has obvious benefits for both the user and the developer. It gives the user the flexibility to perform inversion and analysis tasks for a variety of potentially coupled simulation models using a common, established concept and a single set of instructions. The non-intrusive coupling between the optimization routines and application models allows the developer to focus on improving the inversion and analysis tools rather than on integrating new or modified forward models into the framework.

On the other hand, a tight integration of the simulation and optimization codes (the approach followed by the original iTOUGH2 code) has also its advantages. Sharing variables in memory rather than transferring them through external text files eliminates concerns about the loss of precision, an issue that needs to be carefully addressed when using the PEST protocol. Moreover, fully integrating the simulator into the optimization code allows the latter to be "knowledgeable" about the parameters, observable variables, and the processes being simulated. Input can be streamlined and checked, and the execution of the forward simulation can be controlled and adjusted based on the needs of the inversion.

The tradeoff between flexibility on the one hand and control and convenience on the other is not resolvable without considering a specific application. The extension of iTOUGH2 by including the PEST protocol is intended to provide the user with improved means to solve challenging simulation-optimization problems, using a variety of codes, including TOUGH2-related simulators that have not yet been specifically integrated into the iTOUGH2 framework.

Acknowledgment

We would like to thank John Doherty for making the PEST protocol and related parsing routines publicly available, and to Nicolas Spycher for his review of the manuscript and for providing the TOUGHREACT simulation for the analysis in Section 3.1. We very much appreciate the thoughtful and constructive comments of the three anonymous reviewers. This work was supported, in part, by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Wind and Geothermal Technologies, of the U.S. Department of Energy, and as part of the Subsurface Science Scientific Focus Area funded by the U.S. Department of Energy, Office of Science, Office of Biological and Environmental Resources under Award Number DE-AC02-05CH11231.

References

Ayvaz, M. T., 2007. Simultaneous determination of aquifer parameters and zone structures with fuzzy c-means clustering and meta-heuristic harmony search algorithm. Advances in Water Resources 30, 2326–2338.

Banta, E.R., Hill, M.C., Poeter, E. Doherty, J.E., Babendreier, J., 2008. Building model analysis applications with the Joint Universal Parameter IdenTification and Evaluation of Reliability (JUPITER) API. Computers and Geosciences 34, 310–319.

Bonnet, E., Bour, O., Odling, N.E., Davy, P., Main, I., Cowie, P., Berkowitz, B., 2001. Scaling of fracture systems in geological media. Rev. Geophys 39(3):347–383.

Carrera, J., 1984. Estimation of aquifer parameters under transient and steady-state conditions. Ph.D. dissertation, Department of Hydrology and Water Resources, University of Arizona, Tucson, Arizona.

Carrera, J., Neuman, S.P., 1986. Estimation of aquifer parameters under transient and steady state conditions: 1. Maximum likelihood method incorporating prior information. Water Resources Research 22(2), 199–210.

Doherty, J., 2007. FORTRAN 90 Modules for Implementation of Parallelised, Model-Independent, Model-Based Processing. Watermark Numerical Computing, Australia.

Doherty, J., 2008. PEST: Model-Independent Parameter Estimation. Watermark Numerical Computing, Brisbane, Australia.

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

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

Finsterle, S., Parallelization of iTOUGH2 Using PVM, 1998. Report LBNL-42261, Lawrence Berkeley National Laboratory: Berkeley, Calif.

Finsterle, S., 2000. Using the continuum approach to model unsaturated flow in fractured rock. Water Resources Research 36(8), 2055–2066.

Finsterle, S., 2004. Multiphase inversion modeling: Review and iTOUGH2 applications. Vadose Zone Journal 3, 747–762.

Finsterle, S., 2005. Demonstration of optimization techniques for groundwater plume remediation using iTOUGH2. Environmental Modelling and Software 21(5), 665–680, doi:10.1016/j.envsoft.2004.11.012.

Finsterle, S., Najita, J., 1998. Robust estimation of hydrogeologic model parameters. Water Resources Research 34(11), 2939–2947.

Finsterle, S., Fabryka-Martin, J.T., Wang, J.S.Y., 2002. Migration of a water pulse through fractured porous media. Journal of Contaminant Hydrology 54 (1–2), 37–57.

Finsterle, S., Kowalsky, M.B. 2008. Joint hydrological-geophysical inversion for soil structure identification. Vadose Zone Journal 7, 287–293, doi:10.2136/vzj2006.0078.

Finsterle, S., Ahlers, C.F., Trautz, R.C., Cook P.J., 2003. Inverse and predictive modeling of seepage into underground openings. Journal of Contaminant Hydrology 62–63, 89–109.

Finsterle, S., Doughty, C., Kowalsky, M.B., Moridis, G.J., Pan, L., Xu, T., Zhang, Y., Pruess, K., 2008. Advanced vadose zone simulations using TOUGH. Vadose Zone Journal 7, 601–609, doi:10.2136/vzj2007.0059.

Finsterle, S., Kowalsky, M.B., 2011. A truncated Levenberg-Marquardt algorithm for the calibration of highly parameterized models. Computers and Geosciences, doi:10.1016/j.cageo.2010.11.005.

Finsterle, S., Zhang, Y., 2011. Error handling strategies in multiphase inverse modeling. Computers and Geosciences, doi:10.1016/j.cageo.2010.11.009.

Fujita, Y., Ferris, F.G., Lawson, R.D., Colwell, F.S., Smith, R.W., 2000. Calcium carbonate precipitation by ureolytic subsurface bacteria. Geomicrobiology Journal 17, 305–318.

Fujita, Y., Redden, G.D., Ingram, J.C., Cortez, M.M., Ferris, F.G., Smith R.W., 2004. Strontium incorporation into calcite generated by bacterial ureolysis. Geochim. Cosmochim Acta 68, 3261–3270.

Gallagher, P.M., Finsterle, S., 2004. Physical and numerical model of colloidal silica injection for passive site stabilization. Vadose Zone Journal 3, 917–925.

Gauss, C.F., 1821. Theoria Combinationis Observationum Erroribus Minimis Obnoxiae. In: Stewart, G.W. (Ed.), Theory of the Combination of Observations Least Subject to Errors. SIAM, Philadelphia, PA (translator).

Geem, Z.W., Kim, J.-H., Loganathan, G.V., 2001. A new heuristic optimization algorithm: harmony search. Simulation 76(2), 60–68.

Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., Sunderam, V., 1994. PVM: Parallel Virtual Machine—a User's Guide and Tutorial for Networked Parallel Computing. MIT Press, Cambridge.

Ghezzehei, T.A., Trautz, R.C, Finsterle, S., Cook, P.J., Ahlers, C.F., 2004. Modeling coupled evaporation and seepage in ventilated tunnels. Vadose Zone Journal 3, 806–818.

Jung, Y., Imhoff, P., Finsterle, S., 2011. Estimation of landfill gas generation rate and gas permeability field of refuse using inverse modeling. Transport in Porous Media. doi:10.1007/s11242-010-9659-8.

Kiryukhin, A.V., Asaulova, N.P., Finsterle, S., 2008. Inverse modeling and forecasting for the exploitation of the Pauzhetsky geothermal field, Kamchatka, Russia. Geothermics 37, 540–562, doi:10.1016/j.geothermics.2008.04.003.

Kitterød, N.-O., Finsterle, S., 2004. Simulating unsaturated flow fields based on saturation measurements. Journal of Hydraulic Research 42, 121–129.

Kowalsky, M.B., Finsterle, S., Rubin, Y., 2004. Estimating flow parameter distributions using ground-penetrating radar and hydrological measurements during transient flow in the vadose zone. Advances in Water Resources 27(6), 583–599.

Kowalsky, M., Finsterle, S., Peterson, J., Hubbard, S., Rubin, Y., Majer, E., Ward, A. Gee, G., 2005. Estimation of field-scale soil hydraulic parameters and dielectric parameters through joint inversion of GPR and hydrological data. Water Resources Research 41, W11425, doi:10.1029/2005WR004237.

Kowalsky, M.B., Gasperikova, E., Finsterle, S., Watson, D., Hubbard, S.S., 2011. Coupled modeling of hydrogeochemical and electrical resistivity data for exploring the impact of recharge on subsurface contamination. Water Resources Research 47, W02509, doi:10.1029/2009WR008947.

Levenberg, K., 1944. A method for the solution of certain nonlinear problems in least squares. The Quarterly of Applied Mathematics 2, 164–168.

Linde, N., Finsterle, S., Hubbard, S., 2006. Inversion of tracer test data using tomographic constraints Water Resources Research 42(4), W04410, doi:10.1029/2004WR003806.

Liu, H. H., Bodvarsson, G. S., Finsterle, S., 2002. A note on unsaturated flow in two-dimensional fracture networks. Water Resources Research 38(9), 1176, doi:10.1027/2001WR000977.

Marquardt, D.W., 1963. An algorithm for least squares estimation of nonlinear parameters. SIAM Journal of Applied Mathematics 11, 431–441.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E., 1953. Equation of state calculations by fast computing machines. Journal of Chemical Physics 21, 1087–1092.

Mitchell, A.C., Ferris, F.G., 2005. The coprecipitation of Sr into calcite precipitates induced by bacterial ureolysis in artificial groundwater: Temperature and kinetic dependence. Geochim. Cosmochim. Acta 69, 4199–4210.

Moridis, G.J., Finsterle, S., Heiser, J., 1999. Evaluation of alternative designs for an injectable barrier at the Brookhaven National Laboratory Site, Long Island, New York. Water Resources Research 35(10), 2937–2953.

Nelder, J.A., Mead, R., 1965. A simplex method for function minimization. Computer Journal 7, 307–313.

Poeter, E.P., Hill, M.C., 1998. Documentation of UCODE, a Computer Code for Universal Inverse Modeling. U.S. Geological Survey Water-Resources Investigations Report 98-4080.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 1992. Numerical Recipes in FORTRAN: The Art of Scientific Computing. 2nd edition. Cambridge University Press, Cambridge.

Pruess, K., C. Oldenburg, G. Moridis, 1999. TOUGH2 User's Guide, Version 2.0. Report LBNL-43134, Lawrence Berkeley National Laboratory, Berkeley, Calif.

Spycher, N., Zhang, G., Sengor, S., Issarangkun, M., Barkouki, T., Ginn, T., Wu, Y., Smith, R., Hubbard, S., Fujita, Y., Sani, R., Peyton, B., 2009. Application of TOUGHREACT V2.0 to environmental systems. Proceedings, TOUGH Symposium 2009, Lawrence Berkeley National Laboratory, Berkeley, Calif., September 14–16, 2009.

Storn, R., Price, K., 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359.

Tonkin, M., Doherty, J., 2005. A hybrid regularized inversion methodology for highly parameterized models. Water Resources Research 41, W10412, doi:10.1029/2005WR003995.

Unger, A., Finsterle, S., Bodvarsson, G.S., 2004. Transport of radon gas into a tunnel at Yucca Mountain—estimating large-scale fractured tuff hydraulic properties and implications for the ventilation system. Journal of Contaminant Hydrology 70, 152–171.

Wang, J.S.Y., Trautz, R.C., Cook, P.J., Finsterle, S., James, A.L., Birkholzer, J., 1999. Field tests and model analyses of seepage into drift. Journal of Contaminant Hydrology 38 (1–3), 323–347.

Wu, Y., Ajo-Franklin, J., Spycher, N., Zhang, G., Hubbard, S., Williams, K., Taylor, J., Fujita Y., Smith R., (submitted). Ureolytic calcite precipitation: Geophysical and geochemical responses based on laboratory experiments, synchrotron imaging, and reactive transport modeling.

Xu, T., Sonnenthal, E.L., Spycher, N., Pruess, K., 2004. TOUGHREACT User's Guide: A Simulation Program for Non-Isothermal Multiphase Reactive Geochemical Transport in Variably Saturated Geologic Media. Report LBNL-55460, Lawrence Berkeley National Laboratory, Berkeley, Calif.

Zhang, Y., Pinder, G., 2003. Latin hypercube lattice sample selection strategy for correlated random hydraulic conductivity fields. Water Resources Research 39(8), 1226, doi:10.1029/2002WR001822.

Zhang, Y., Freifeld, B., Finsterle, S., Leahy, M., Ennis-King, J., Paterson, L., Dance, T., 2011. Single-well experimental design for studying residual trapping of supercritical carbon dioxide. International Journal of Greenhouse Gas Control 5, 88–98, doi:10.1016/j.ijggc.2010.06.011.

Zhang, Y., Hubbard, S. Finsterle, S., 2011. A numerical study on sustainable pumping near rivers Ground Water, doi: 10.1111/j.1745-6584.2010.00743.x.

Table 1. Minimization algorithms implemented in iTOUGH2

Minimization Algorithm Reference	Description	Comments
	Local Search Algorithms	
Downhill Simplex (Nelder and Mead, 1965)	Approaches minimum through sequence of reflections, expansions, and contractions of an $(n+1)$-dimensional simplex.	No assumptions made about form of cost function; relatively inefficient.
Gauss-Newton (Gauss, 1821)	$\Delta\mathbf{p} = \left(\mathbf{J}^T\mathbf{W}\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}$ Includes truncated SVD	Efficient for linear least-squares problems only; requires derivatives.
Levenberg-Marquardt (Marquardt, 1963; Levenberg, 1944; Finsterle and Kowalsky, 2011)	$\Delta\mathbf{p} = \left(\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\mathbf{D}\right)^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}$ Includes different Tikhonov matrices and automatic truncation	Efficient for nonlinear least-squares problem; requires derivatives.
	Global Search Algorithms	
Grid Search	Evaluate cost function in entire parameter space on regular or irregular grid.	Complete information about cost function; very inefficient.
Simulated Annealing (Metropolis et al., 1953)	Metaheuristic algorithm mimicking slow cooling of metals; includes thermal fluctuations and temperature schedule; accepts uphill steps based on Metropolis algorithm with decreasing probability.	No assumptions made about cost function; may escape local minima; inefficient.
Harmony Search (Geem et al., 2001; Ayvaz, 2007)	Metaheuristic algorithm mimicking musical improvisation; searches for harmony by improvisation and pitch adjustment.	
Differential Evolutionary Algorithm (Storn and Price, 1997)	Metaheuristic algorithm mimicking evolution with weighted differences between populations and trial vector.	

[a] **J** Jacobian matrix; **W=**: Weighting matrix; **D**: Tikhonov matrix; **r**: Residual/cost vector

Table 2. iTOUGH2 and PPEST Inversion Results of TOUGHREACT Model

Parameter	Initial	Best Estimate		Uncertainty	
		PPEST	iTOUGH2	PPEST	iTOUGH2
Obj. Function	7.3034	5.2356	5.2356	n/a	n/a
Model Runs	n/a	91	51	n/a	n/a
Init. enzyme conc.	3.000×10^{-10}	1.847×10^{-10}	1.845×10^{-10}	0.239×10^{-10}	0.238×10^{-10}
Init. biomass conc.	1.022×10^{-14}	1.026×10^{-14}	1.026×10^{-14}	0.092×10^{-14}	0.093×10^{-14}
$\log_{10}$(precip. rate)	-7.398	-7.302	-7.304	0.051	0.055
K selectivity	0.49	0.538	0.535	0.129	0.125
Cation exch. cap.	10.000	8.608	8.580	1.148	1.090

n/a: not applicable

Table 3. Steps to Generate Discrete Fracture Network Model

Step	Activity	Software
0	Script file invoking mesh generation steps 1–6 External executable called before each TOUGH2 simulation	*sh.DFNMgen*
1	Generate 2D network of fracture traces based on statistical parameters on fracture density, fracture length, and fracture orientation provided through an input file that is created by the PEST template file Remove unconnected fractures Discretize fracture traces, assign aperture and permeabilities to fracture elements, create TOUGH2 element and connection information	*xDFNM*
2	Combine element and connection information block to create base mesh file	*sh.DFNMgen*
3	Move X and Z coordinates of mesh	*xMoveMesh8*
4	Add top boundary element	*xAddBound8*
5	Add bottom boundary element	*xAddBound8*
6	Cut out niche from mesh, adjust permeabilities near niche to reflect excavation disturbed zone	*xCutNiche8*

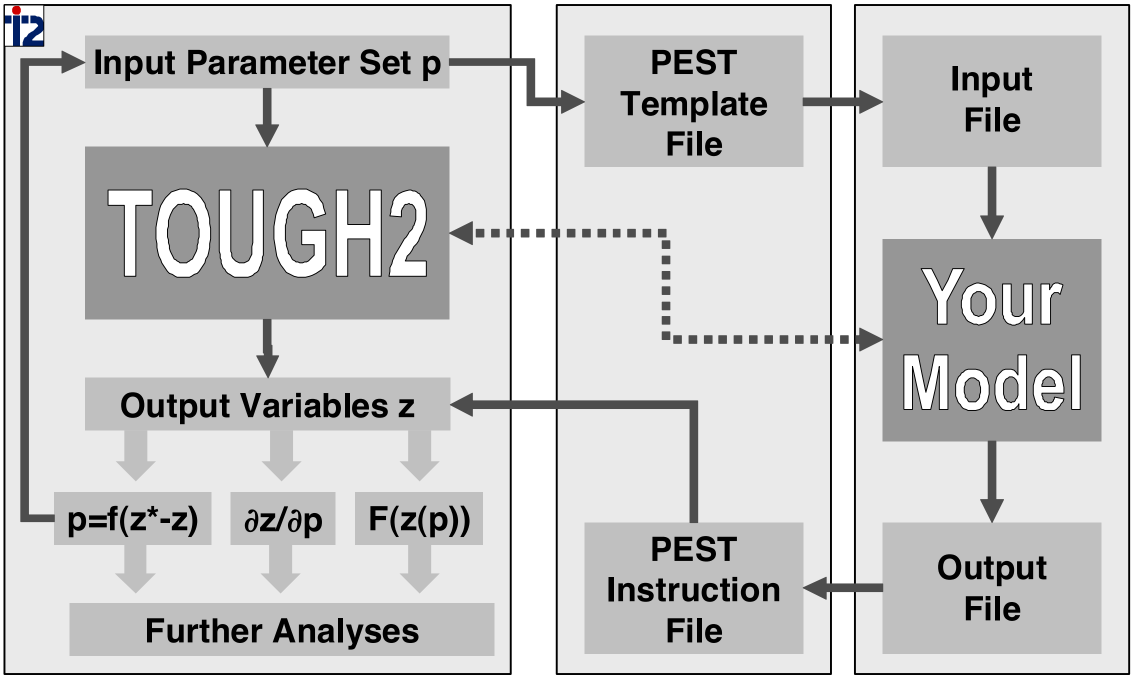


Figure 1. iTOUGH2 architecture: optimization and analysis tools evaluate the system response **z** as a function of adjustable input parameters **p**, where the relation between **z** and **p** is either given by the fully integrated TOUGH2 simulator or by an external model through the PEST protocol, which uses text-based template and instruction files for communication with the external model, which is shown in the right-most box.

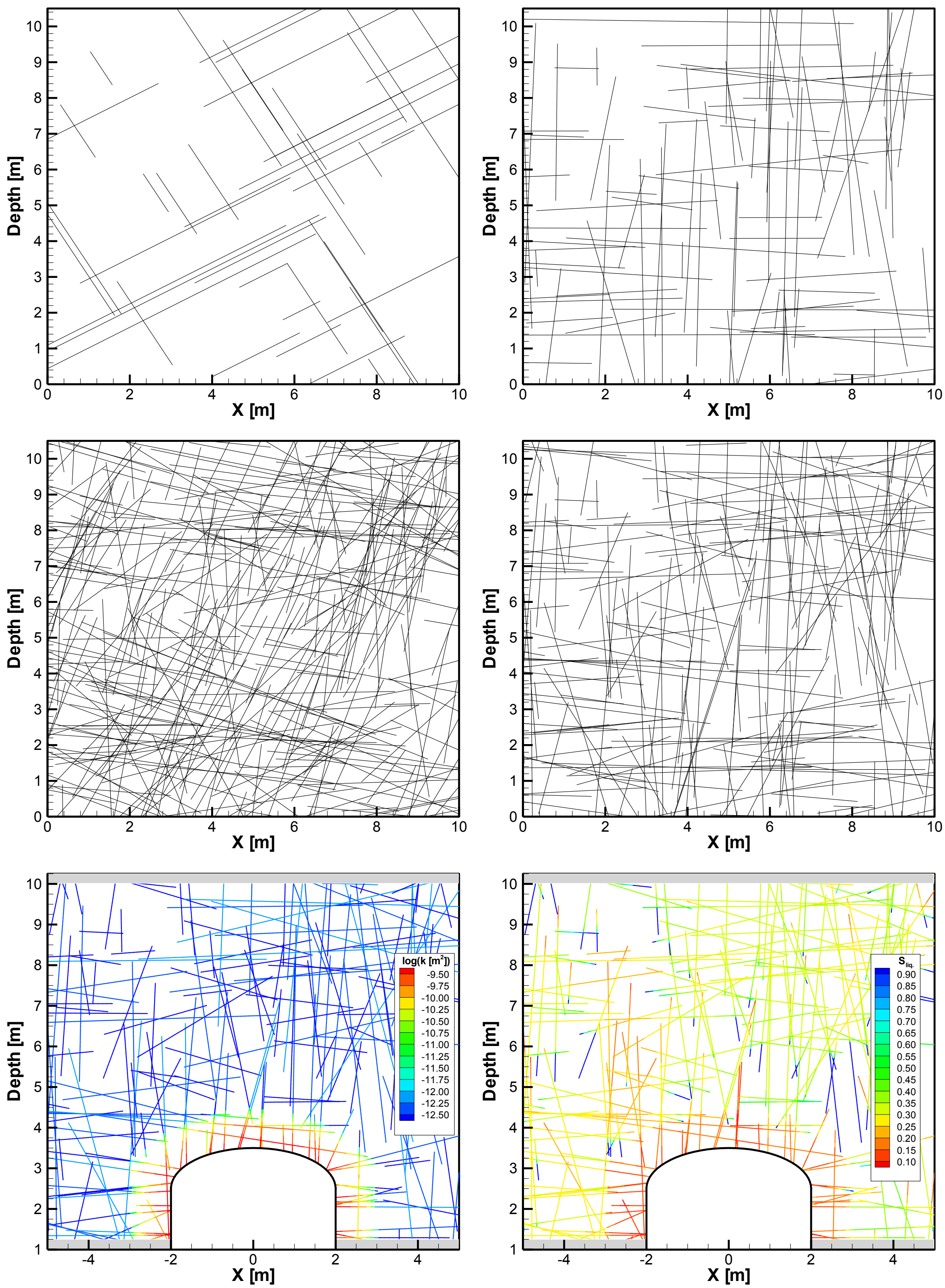
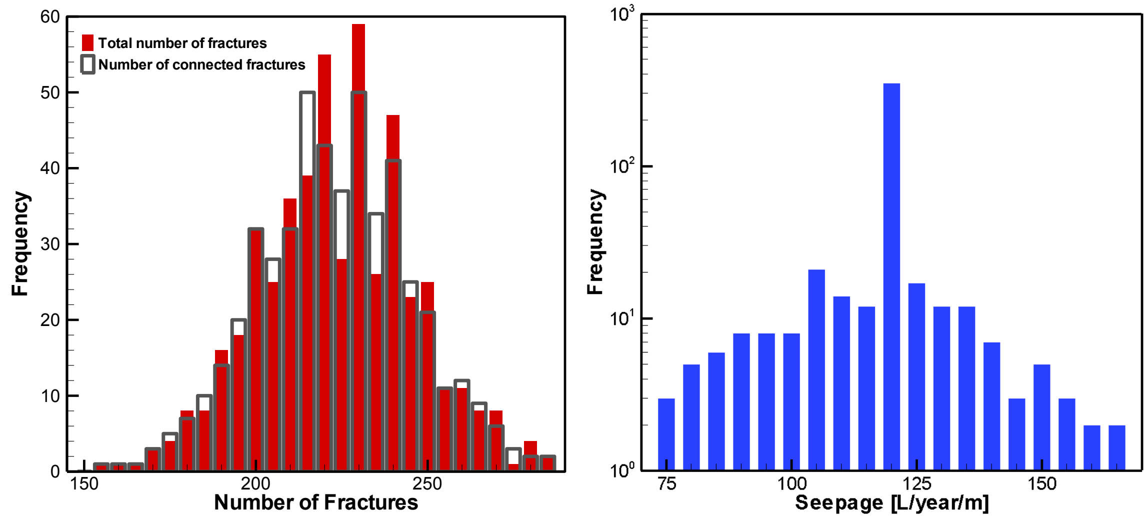


Figure 2. Four realizations of the base discrete fracture network, permeability field, and steady-state saturation distribution.

```
> COMPUTATION

  >> STOP
     >>> Number of SIMULATIONS: 500
     <<<

  >> ERROR propagation analysis
     >>> MONTE CARLO SEED: 5555
     >>> LATIN HYPERCUBE SAMPLING CORRELATION MATRIX: 6
         1E-4    0.0    0.0    0.0   0.0   0.0
          0.0  100.0    0.0    0.0   0.0   0.0
          0.0    0.0  100.0    0.0   0.9   0.0
          0.0    0.0    0.0    4.0   0.0   0.5
          0.0    0.0    0.9    0.0 100.0   0.0
          0.0    0.0    0.0    0.5   0.0   4.0
     <<<

  >> OPTION
     >>> PEST
         >>>> EXECUTABLE     : sh.DFNMgen     run BEFORE TOUGH2!
         >>>> TEMPLATE       : 1
                               input.tpl      input.dat
         >>>> INSTRUCTION    : 1
                               fracture.ins   fracture.frq
         <<<<

     >>> STEADY STATE

     >>> PVM: 30  FILE: NODEFILE
     HOST1PVM
     HOST2PVM
     ...
     HOST30PVM
     <<<
  <<
<
```

Figure 3. Excerpt of iTOUGH2 input file with control parameters.



(a) (b)

Figure 4. (a) Histogram of number of fractures generated for different statistical input parameters, and (b) resulting distribution of annual seepage per meter of tunnel.

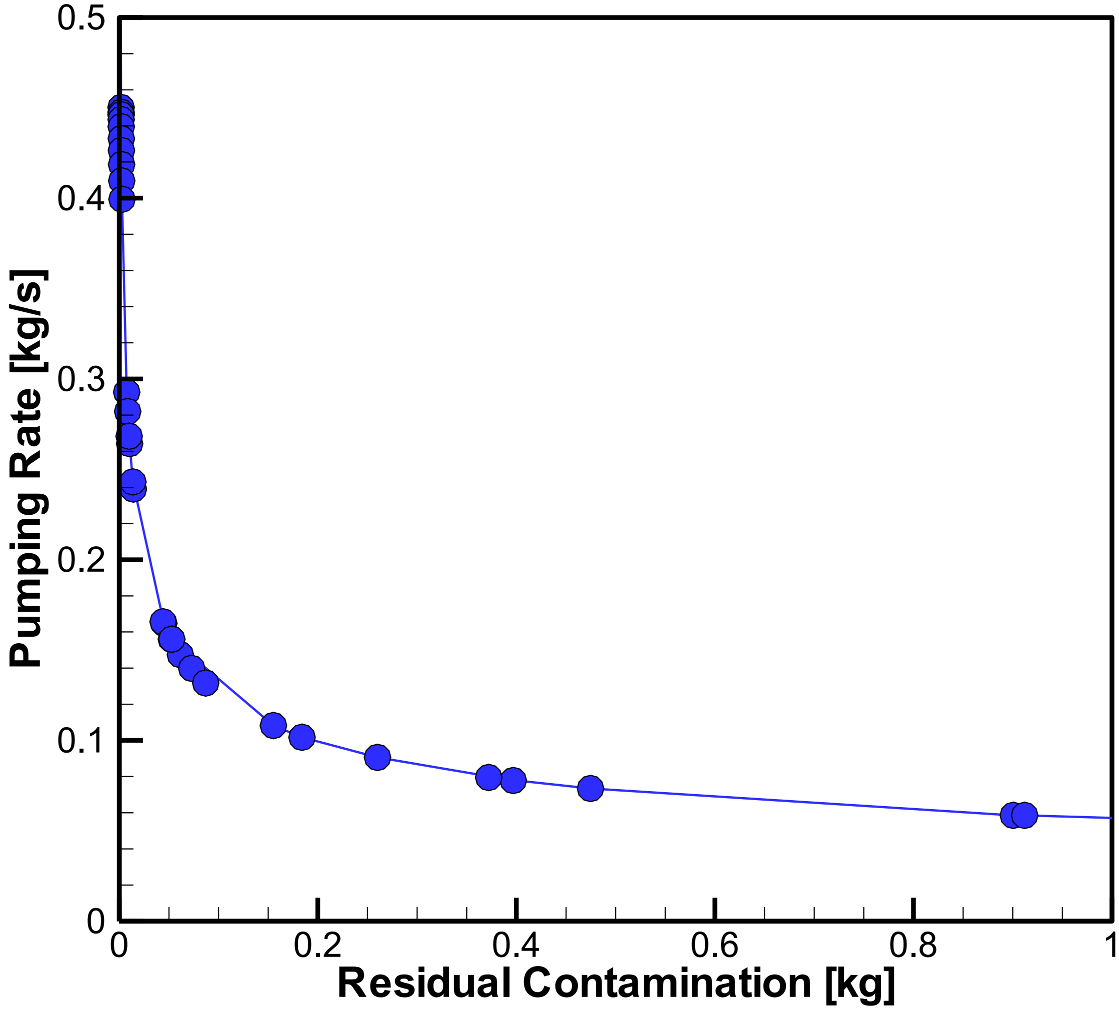


Figure 5. Pareto frontier.

***Suggested Reviewer List (include up to 5 names and their contact details)**

John Doherty, jdoherty@gil.com.au
Michael O'Sullivan, m.osullivan@auckland.ac.nz
Rainer Senger, rsenger@intera.com
Florian Wellmann, wellmann@cyllene.uwa.edu.au

DISCLAIMER