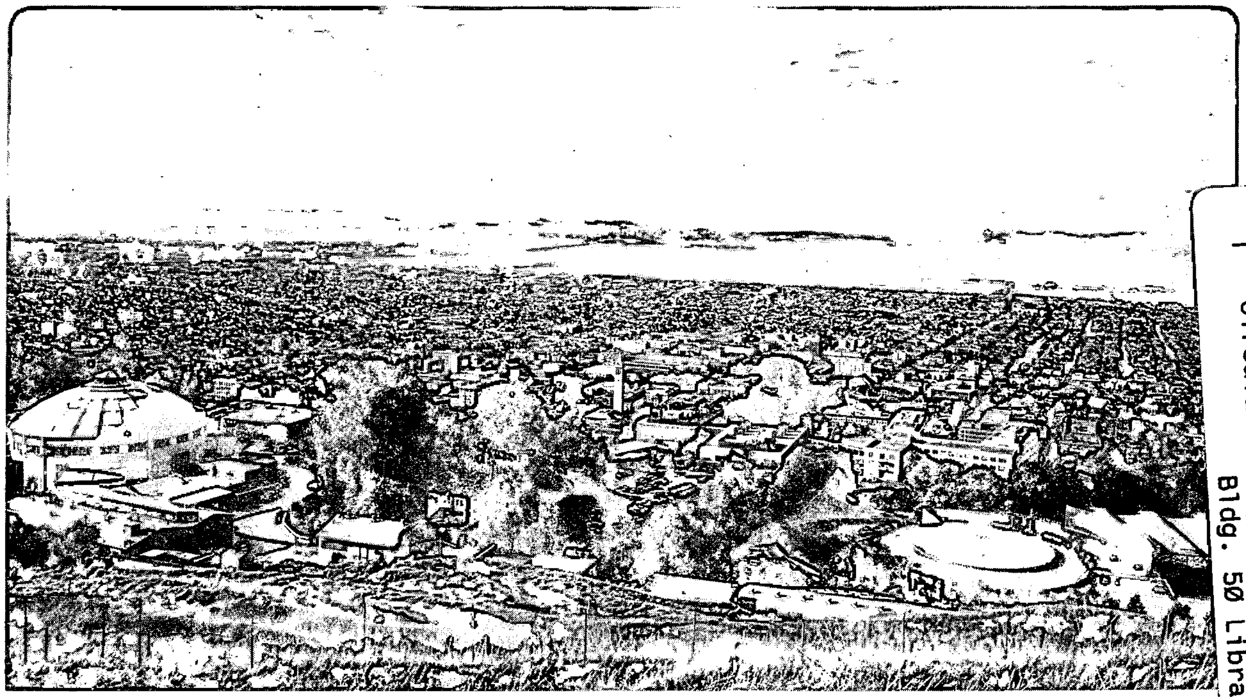
LBL-34581
UC-600

EARTH SCIENCES DIVISION

ITOUGH2 User's Guide Version 2.2

S. Finsterle

August 1993



Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

LBL-34581

ITOUGH2 User's Guide

Version 2.2

Stefan Finsterle

Earth Sciences Division
Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

August 1993

This work was carried out under U.S. Department of Energy Contract No. DE-AC03-76SF00098 for the Director, Office of Civilian Radioactive Waste Management, Office of External Relations, and was administered by the Nevada Operations Office, U.S. Department of Energy, in cooperation with the Swiss National Cooperative for the Disposal of Radioactive Waste (Nagra), and was supported, in part, by the WIPP project, Sandia National Laboratories, under Document No. 129847.

Abstract

ITOUGH2 is a program to estimate hydrogeologic model parameters for the numerical simulator TOUGH2.

TOUGH2 was developed by Karsten Pruess at Lawrence Berkeley Laboratory for simulating non-isothermal flows of multicomponent, multiphase fluids in porous and fractured media.

ITOUGH2 solves the inverse problem by automatic model calibration based on an indirect approach, in which some function of the difference between observed and model-predicted system response and appropriately weighted prior information about the parameters is minimized using standard optimization techniques. ITOUGH2 also provides a detailed error analysis of the estimated parameter set, and employs some procedures to study error propagation for prediction runs.

This report includes a review of the inverse modeling theory, and a detailed description of the program architecture, input language, and the various user features provided by ITOUGH2. A sample problem is given to illustrate code application.

Table of Contents

List of Figures

List of Tables

1 . Introduction

Groundwater systems are often analyzed using mathematical models which are solved by means of computer simulations. While standard groundwater models are designed to predict the behavior of an aquifer for given initial and boundary conditions, *inverse modeling* deals with the question of how to appropriately assign values to the various model parameters. This report describes the theoretical background, the program architecture, input language, and user features of the ITOUGH2 code which solves the inverse problem for the two-phase two-component numerical simulator TOUGH2 [*Pruess*, 1991].

Inverse modeling consists of estimating model parameters from measurements of the system response made at discrete points in space and time. The parameters to be estimated are the coefficients in the governing flow equations which represent the hydrogeological properties of the aquifer. Their interpretation depends on the model structure and the purpose the specific model is intended for. In this sense, the parameters are strictly to be seen as *model* parameters rather than *aquifer* parameters. Estimating parameter values from measurements therefore relates the real groundwater system to its representation in a physically based mathematical model.

Inverse modeling involves several interacting steps. Starting from a conceptual model of the physical system, the results of the parameter estimation may indicate that the underlying model structure needs some modification. While inverse modeling may be discussed in the broader context of identification, ITOUGH2 only solves the parameter estimation problem for a given model structure.

Parameter estimation covers both data collection and model development. The necessary steps can be summarized as follows:

- Information about the model parameters is drawn from measurements of the system state at discrete points in space and time. Good data in terms of quantity and quality are the key condition inverse modeling is based on. Therefore, type, location, and duration of hydraulic tests, the data acquisition system, as well as processing the quantities being measured have to be carefully designed in order to obtain as much sensitive data as possible. In addition, one should be able to quantify the measurement and interpretation errors.

- Inverse modeling starts with the formulation of the so-called *direct problem.* A site-specific model has to be developed which is capable of simulating the *general* hydraulic situation of the groundwater system under measurement conditions. This step involves the description of the relevant physical processes, the definition of model geometry, assigning types of boundary conditions, discretizing the problem in space and time, selecting zones over which the model parameters are believed to be constant, etc. All the parameters that are not subject to the estimation process are then fixed at their best known values. It is important to realize that these fixed parameters are part of the model structure to which the solution of the inverse problem will refer.

  The direct problem is solved by the TOUGH2 simulator, written by Karsten Pruess at Lawrence Berkeley Laboratory. Experience in using TOUGH2 is an essential requirement for running ITOUGH2. The preparation of a TOUGH2 input deck is described in *Pruess* [1987, 1991] and will be taken for granted.

- The next step is to define a vector containing all parameters for which a numerical value is to be determined. An initial guess for each of the parameters has to be assigned and appropriately weighted. Some of the parameters may have to be transformed (e.g. estimate logarithm instead of parameter value itself).

- Finally, ITOUGH2 provides a procedure that relates the measured data to the unknown model parameters. A number of quantities describing uncertainties give some insight to the quality of the estimated parameter set.

A summary description of the TOUGH2 code is given in the following Chapter. However, no details are presented in this manual. The inverse problem can be posed within the framework of maximum-likelihood estimation; the theoretical background is reviewed in Chapter 3 including a brief description of the minimization algorithm and the error analysis. In Chapter 4, the program architecture and the preparation of an ITOUGH2 input file is described. Finally, a detailed example is discussed in Chapter 5.

2. The Direct Problem: Summary Description of TOUGH2 Simulator

TOUGH2 is a numerical simulation program for multi-dimensional coupled fluid and heat flows of multiphase multicomponent fluid mixtures in porous and fractured media [*Pruess*, 1991]. TOUGH2 is a more general version of the TOUGH simulator [*Pruess*, 1987]. The main extension consists of a number of fluid property modules and enhanced user features such as internal mesh generation. The coupled transport of multiple components in multiple phases is calculated by means of integrated finite differences. Fluid flow occurs under pressure, viscous, and gravity forces according to Darcy's law, with interferences between the phases represented by predefined or user specified relative permeability functions. Capillary forces are given as nonlinear functions of liquid saturation. In addition, binary diffusion is considered in the gas phase. Thermophysical properties of liquid water and vapor are taken from steam table equations. The gaseous phase is treated as ideal, and additivity of partial pressures is assumed for air/vapor mixtures. Dissolution in the liquid phase is represented by Henry's law. Heat transport occurs by means of conduction, with thermal conductivity dependent on liquid saturation, and convection and binary diffusion, which includes both sensible and latent heat.

A detailed description of the physical processes, the governing equations, the numerical methods, and the preparation of a TOUGH2 input file is given by *Pruess* [1987, 1991]. Since TOUGH2 is used to solve the direct problem, the parameters to be estimated are related to the model structure given by the code and the specific model of the system. Therefore, only TOUGH2 input parameters may be estimated based on observations for which TOUGH2 calculates a corresponding output. Furthermore, certain TOUGH2 typing conventions have to be obeyed.

It should be realized that the TOUGH2 model conceptualization is the most important step when performing inverse modeling because a consistent and stable solution of the direct problem is the basis on which the estimation of the model parameters is built.

3. Inverse Modeling Theory

3.1 Introduction

The inverse problem of parameter estimation can be formulated as an optimization procedure where the objective is to minimize some norm of the difference between observed and model predicted system response and appropriately weighted prior information about the aquifer properties. The model parameters can be viewed as a set of deterministic quantities which are uncertain due to insufficient data and their corruption by noise. This viewpoint leads to the maximum likelihood (ML) approach in which one maximizes the probability of observing the measured data thus leading to the parameter set most likely to be true.

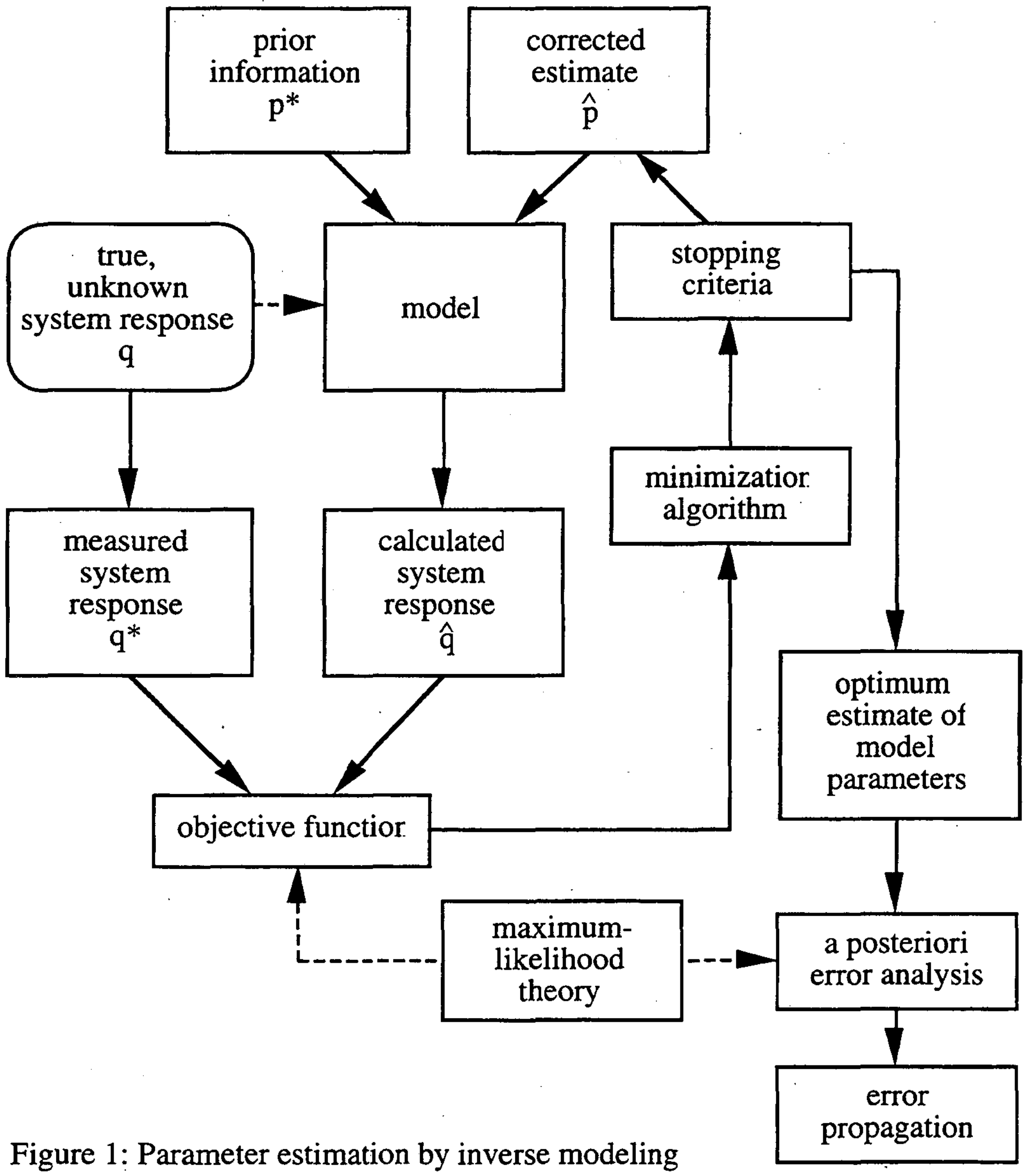

prior information p*
corrected estimate p̂
true, unknown system response q
model
stopping criteria
minimization algorithm
measured system response q*
calculated system response q̂
optimum estimate of model parameters
objective function
maximum-likelihood theory
a posteriori error analysis
error propagation

Figure 1: Parameter estimation by inverse modeling

Figure 1 shows the concept of parameter estimation by inverse modeling. The true, but unknown system behavior is observed at discrete points in space and time. The aquifer is also modeled using TOUGH2 which calculates the system response as a function of the model parameters. Maximum likelihood theory provides an objective function which has to be minimized by means of a standard minimization algorithm. The model parameters are iteratively updated until an optimum parameter set is determined. A posteriori error analysis is performed to assess the quality of the estimates. The impact of parameter uncertainty on the model prediction may be studied in a subsequent step.

The likelihood concept is briefly introduced in the next Section. The algorithm which minimizes the objective function is described in Section 3.4; a simple linear error analysis is outlined in Section 3.5, followed by a description of two methods to calculate the prediction error. A more detailed discussion can be found in *Carrera* [1984], *Carrera & Neuman* [1986], and *Finsterle* [1993].

3.2 Definitions

Let **p** be the n-dimensional vector of model parameters to be estimated where n is the number of unknowns. The parameter vector **p** may hold values of the absolute permeability, porosity, rock compressibility, various parameters of the relative permeability and capillary pressure functions, initial or boundary pressure, temperature, or gas saturation, fracture spacing, production rates, etc. A detailed list of all parameters which can be estimated is given in Section 4.3.3. The system state (pressure, gas saturation, and temperature distribution, gas and liquid flow rate, etc.) is a function of **p**. The set of discrete observations made of the system response are represented by the vector **q** of dimension m. Let **z** be the vector of measurable variables including model parameters **p** and observations **q**. Each component of **z** will be assumed to have a true value, z, a measured value z*, and a computed value $\hat{z}(\mathbf{p})$. The *measurement error* is defined as the difference between measured and true value (z* - z); the *model* or *computation error* is the difference between the true and the computed value (z - $\hat{z}$). Since the true values of the variables are not known, only the residuals r - the sum of the computation and measurement errors - can be evaluated:

$$\mathbf{r} = (\mathbf{z}^* - \hat{\mathbf{z}}) = (\mathbf{z}^* - \mathbf{z}) + (\mathbf{z} - \hat{\mathbf{z}}(\mathbf{p})) \tag{1}$$

Since the residuals are a sum of errors, the residuals **r** can be treated as a random variable. Based on the central limit theorem, it is reasonable to assume that the residuals are normally distributed with zero mean and a covariance matrix **C**. Furthermore, it is assumed that the error structure of the residuals is *a priori* given, up to an unknown factor. Therefore, **C** can be written as a product of a positive definite symmetric matrix **V** and a scalar σ_0^2:

$$\mathbf{C} = \sigma_0^2 \cdot \mathbf{V} \tag{2}$$

If the covariance matrix of the residuals is believed to be well known, the statistical parameter σ_0^2 can be set to a fixed, arbitrarily chosen value. The *a posteriori* error analysis provides an estimate of $\hat{\sigma}_0^2$ on the basis of the existing data. If $\hat{\sigma}_0^2$ deviates significantly from the *a priori* value σ_0^2, then there is an inconsistency in **V** or - more likely - in the model structure.

It is assumed that the covariance matrix **C** has a block diagonal structure where the submatrices $\mathbf{C}_i$ $i \in \{p,pre,flow,temp,sat\}$ represent the covariances of the parameter, pressure, flow rate, temperature, and saturation measurements, respectively.

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_p & 0 & 0 & 0 & 0 \\ 0 & \mathbf{C}_{pre} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{C}_{flow} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{C}_{temp} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{C}_{sat} \end{bmatrix} \tag{3}$$

This assumes that observations of different types are not correlated. However, the submatrices $\mathbf{C}_i$ may contain non-zero off-diagonal elements reflecting correlated random error components. Each submatrix $\mathbf{C}_i$ has its own factor $(\sigma_0^2)_i$ $i \in \{p,pre,flow,temp,sat\}$; they may be either fixed or treated as unknown parameters which are estimated simultaneously with the model parameters. The procedure is described in Section 4.3.5.4.

In summary: inverse modeling provides estimates of the model parameters $\hat{\mathbf{p}}$. They are determined based on discrete observations $\mathbf{q}^*$ made on the system response and prior information $\mathbf{p}^*$ about the parameters. The performance measure to be minimized is a function of the residuals **r** - the differences between observed and computed values $(\mathbf{z}^* - \hat{\mathbf{z}})$ - weighted by the inverse of the covariance matrix **C** which contains the statistical parameters describing both the computation and the measurement errors.

3.3 Maximum Likelihood

In a statistical framework, the parameter vector **p** can be seen as a hypothesis regarding the values of the model parameters. Let $f(\mathbf{z}^*|\mathbf{p})$ be the conditional probability density of occurrence of the data $\mathbf{z}^*$ given **p** and a specific model structure. The likelihood function $L(\mathbf{p}|\mathbf{z}^*)$ may then be interpreted as a measure of how the data $\mathbf{z}^*$ support the hypothesis regarding **p**. The set of parameters that maximizes the likelihood function may be considered optimal in the view of the existing data. Provided that the error structure of the residuals can be described by the covariance matrix **C**, then the likelihood of **p** given $\mathbf{z}^*$, $L(\mathbf{p}|\mathbf{z}^*)$, is proportional to $f(\mathbf{z}^*|\mathbf{p})$, and is given by

$$L(\mathbf{p}|\mathbf{z}^*) \sim f(\mathbf{z}^*|\mathbf{p}) = \frac{1}{\sqrt{(2\pi)^M \cdot |\mathbf{C}|}} \cdot e^{\left\{-\frac{1}{2}\left[(\mathbf{z}^* - \mathbf{z})^T\, \mathbf{C}^{-1}\, (\mathbf{z}^* - \mathbf{z})\right]\right\}} \tag{4}$$

where |...| indicates determinant, M is the total number observations of type i (including prior information about parameters) with $M = \sum_i M_i$. Maximizing (4) is usually obtained by minimizing the so-called log-likelihood criterion

$$S = -2 \cdot \ln(L(\mathbf{p}|\mathbf{z}^*)) \tag{5}$$

Substituting (4) into (5) and recalling the block diagonal structure of the matrix **C**, the log-likelihood criterion can be rewritten as

$$S = \sum_i \frac{Z_i}{(\sigma_0^2)_i} + \sum_i \ln\left(|\mathbf{V}_i|\right) + \sum_i M_i \cdot \ln\left((\sigma_0^2)_i\right) + M \cdot \ln(2\pi) \tag{6}$$

where:

$$Z_i = (\mathbf{z}^* - \hat{\mathbf{z}})_i^T \cdot \mathbf{V}_i^{-1} \cdot (\mathbf{z}^* - \hat{\mathbf{z}})_i \tag{7}$$

Again, the index i ($i \in \{p, pre, flow, temp, sat\}$) represents the different types of observations including the prior information about the parameters. Provided that the scalars $(\sigma_0^2)_i$ and the matrices $\mathbf{V}_i$ are known, minimization of (6) is equivalent to minimizing the following objective function:

$$\zeta(\hat{\mathbf{p}}) = \sum_i \frac{1}{(\sigma_0^2)_i} \cdot (\mathbf{z}^* - \hat{\mathbf{z}})_i^T \cdot \mathbf{V}_i^{-1} \cdot (\mathbf{z}^* - \hat{\mathbf{z}})_i \tag{8}$$

Minimizing (8) is known as solving the non-linear least squares problem where ζ is a function of the unknown parameter vector $\hat{\mathbf{p}}$. The Levenberg-Marquardt modification of the Gauss-Newton algorithm to minimize (8) is presented in the next Section.

Recall that the objective function (8) has been derived under the assumption that the error structure of the residuals is Gaussian. However, the errors associated with field data show many more outlier points than one would expect from the tail of the normal distribution. Moreover, a simulation model is only able to reproduce an average trend of the true system behavior due to the incompleteness and inaccuracy of the underlying conceptual model. As a result, the residuals, which contain both model and measurement errors, may have a substantial contribution from deviations which are systematic rather than random, and which cannot be properly described by statistical measures. ITOUGH2 provides a choice of so-called *Robust Estimators* to reduce the impact of outliers on the parameter estimates. If no correlations are present, the vector of the weighted residuals **r** contains elements of the form

$$r_i = \frac{z_i^* - \hat{z}_i}{c_{ii}} \tag{9}$$

Let us define a function ρ of the weighted residuals which is the negative logarithm of the assumed probability density function. Then, the objective function ζ is

$$\zeta = \sum_{i=1}^{M} \rho_i \tag{10}$$

where M is the number of data points including prior information about the parameters. Again, minimizing ζ is equivalent to maximizing the probability of reproducing the observed system state. For normally distributed residuals, we obtain

$$\text{(Least Squares)} \qquad \rho_i = r_i^2 \tag{11}$$

By inserting (11) in (10), we obtain (8) for uncorrelated residuals. If the residuals are distributed as a double exponential, then

$$(L_1\text{-Estimator}) \qquad \rho_i = |r_i| \tag{12}$$

and the corresponding estimator is obtained by minimizing the mean absolute deviation, also termed L_1-estimator. We now assume that the residuals obey a distribution Ω which is defined as follows:

$$\Omega = (1 - \varepsilon)\cdot N + \varepsilon\cdot\Phi \tag{13}$$

Here, N is the Gaussian distribution, Φ is the unknown distribution of the outliers, and ε is the (small) probability with which outliers occur. From (13) it follows that the objective function to be minimized is the sum of squared residuals for all weighted deviations with probability (1-ε) plus an unknown contribution from the remaining large residuals. The latter could be determined from maximum likelihood considerations if Φ were known. We follow the suggestion of *Carosio* [1979] and formulate a function $\rho(r)$ which accounts for the fact that the tail of the distribution Ω is somewhat more prominent than the one of the Gaussian distribution, thus leading to a lower weight of large residuals in the objective function:

$$\text{(Robust Estimator 1)} \qquad \rho_i = \begin{cases} r_i^2 & |r_i| \leq k \\ 2k\cdot|r_i| - k^2 & |r_i| > k \end{cases} \tag{14}$$

The parameter k is a quantile related to ε. Differences between observed and model predicted state variables which are larger than k times the prior standard deviation σ_i are subject to a linear rather than quadratic contribution to the objective function. The slope of the linear function is equal to the slope of the quadratic function at r=k. Another possibility is to discard the weight associated with all deviant points:

$$\text{(Robust Estimator 2)} \qquad \rho_i = \begin{cases} r_i^2 & |r_i| \leq k \\ k^2 & |r_i| > k \end{cases} \tag{15}$$

where the cut-off value k could be prescribed a priori or determined during the optimization procedure as a multiple of the estimated error variance. The different contributions of weighted residuals to the objective function are depicted in Figure 2 for k=1. Note that the two proposed functions (14) and (15) do not correspond to standard probability distributions.

Since the two robust estimators proposed herein are close to the least squares formulation, it is justified to apply the standard optimization algorithms to minimize (10); they will be presented in the next section.

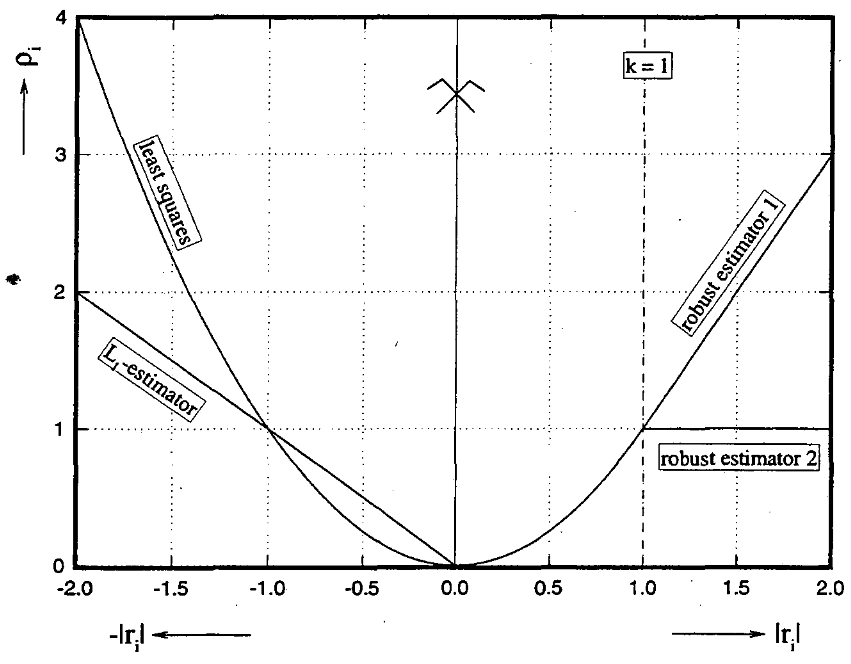


Figure 2: Function ρ_i for least squares, L_1-estimator, and the robust estimators for k=1

3.4 Minimization Algorithm

3.4.1 Levenberg-Marquardt

Because the direct problem is nonlinear in the parameters, the inverse problem has to be solved iteratively. Starting from an initial parameter vector $\mathbf{p}_0$, holding the observed or estimated prior information about the parameters, the procedure involves computing a correction vector $\Delta\mathbf{p}_k$ such that the new estimate

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \Delta\mathbf{p}_k \tag{16}$$

reduces the objective function $\zeta(\mathbf{p}_{k+1}) < \zeta(\mathbf{p}_k)$ at each iteration k. The correction vector $\Delta\mathbf{p}_k$ is calculated using the Levenberg-Marquardt modification of the Gauss-Newton minimization algorithm. Briefly, this procedure involves solving the following nonlinear system of equations:

$$(\mathbf{J}_k^T \mathbf{C}^{-1} \mathbf{J}_k + \mu \cdot \mathbf{D}_k) \cdot \Delta\mathbf{p}_k = - \mathbf{J}_k^T \mathbf{C}^{-1} \mathbf{f}_k \tag{17}$$

where k labels iteration, $\mathbf{f} = (\mathbf{z}^* - \hat{\mathbf{z}})$ is the vector of residuals, **C** is the covariance matrix, **D** denotes a diagonal matrix of order *n* with elements equivalent to the diagonal elements of matrix $\mathbf{J}^T \mathbf{C}^{-1} \mathbf{J}$, μ is a scalare known as the Levenberg-Marquardt parameter, and **J** is the Jacobian matrix with elements given by a forward finite difference approximation:

$$J_{ij} = \frac{\partial \hat{z}_i}{\partial p_j} \approx \frac{\hat{z}_i(p_j+\delta p_j) - \hat{z}_i(p_j)}{\delta p_j} \qquad (18a)$$

or by a centered finite difference quotient:

$$J_{ij} = \frac{\partial \hat{z}_i}{\partial p_j} \approx \frac{\hat{z}_i(p_j+\delta p_j) - \hat{z}_i(p_j-\delta p_j)}{2 \cdot \delta p_j} \qquad (18b)$$

with the difference increment:

$$\delta p_j = \alpha \cdot p_j \qquad (19)$$

where α is a user specified factor (default: 0.01). Recall, that calculating the Jacobian matrix using (18a) requires solving the direct problem (n+1) times, whereas second order accuracy can be obtained solving the direct problem (2n+1) times, and using formula (18b). Obviously, the evaluation of the Jacobian is the most costly part of inverse modeling.

The purpose of the term $\mu \cdot \mathbf{D}_k$ in (17) is to make the approximation of the Hessian matrix ($\mathbf{J}^T\mathbf{C}^{-1}\mathbf{J}$) more positive definite. This is achieved simply by adding an appropriately scaled positive quantity to its diagonal terms. The positive scalar μ controls both step direction and step size. A large value of μ will result in a small step in the steepest descent direction, while μ=0 performs a full Gauss-Newton step leading to the minimum of the quadratic approximation of the objective function. The situation is sketched in Figure 3 for a two-dimensional objective function. The ellipsodal contours represent the approximation of the actual objective function by the term $\mathbf{J}^T\mathbf{C}^{-1}\mathbf{J}$. They are equivalent to the objective function of a model which is linear in the unknown parameters. It should be realized, that the quality of this approximation is only acceptable close to the optimum, where the model can be represented by its linearization. The correction vector $\Delta\mathbf{p}$ is schematically shown as a function of the Levenberg parameter μ (bold curved line). For μ=0, the algorithm detects the minimum of the local approximation calculated at point **p**.

Strategies of how to update μ at each iteration are described in *Marquardt* [1963]. In general, the minimization procedure starts with a relatively large value for μ. As $\hat{\mathbf{p}} \rightarrow \mathbf{p}$, the algorithm proposes $\mu_k \rightarrow 0$ so that the method acquires the asymptotic rate of convergence of the Gauss-Newton method.

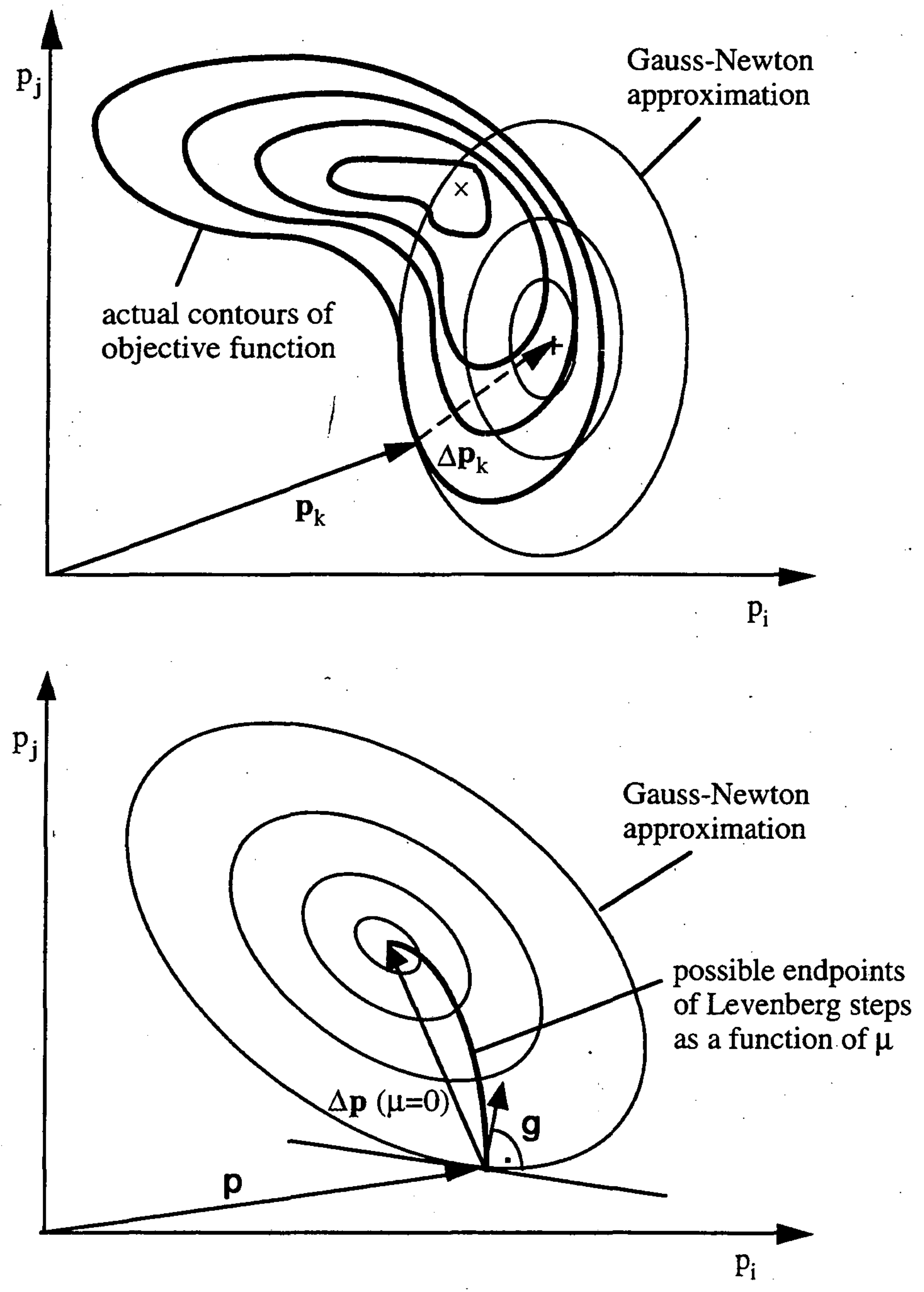

p_j
Gauss-Newton approximation
actual contours of objective function
$\Delta\mathbf{p}_k$
$\mathbf{p}_k$
p_i
p_j
Gauss-Newton approximation
possible endpoints of Levenberg steps as a function of μ
$\Delta\mathbf{p}$ (μ=0)
$\mathbf{g}$
$\mathbf{p}$
p_i

Figure 3: Gauss-Newton approximation and effect of Levenberg-Marquardt parameter on correction vector $\Delta\mathbf{p}$

The local minimum is detected iteratively by solving (17) for $\Delta\mathbf{p}_k$; a new estimate $\mathbf{p}_{k+1}$ is then calculated according to (16). A first stopping criterion may be the number of iterations. The second stopping criterion occurs when either the norm of the objective function or the scaled gradient is less than a given tolerance. The third stopping criterion occurs when the scaled distance between the last two iterations is less than a user supplied step tolerance.

ITOUGH2 also provides a Quasi-Newton algorithm. The routine computes the search direction according to a positive definite approximation of the Hessian $\mathbf{B}_k$, and the gradient evaluated at $\mathbf{p}_k$. A line search is used to find an appropriate step length. When optimality is not achieved, $\mathbf{B}_k$ is updated according to the *Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula* (for details see *Scales* [1985]).

Test runs with ITOUGH2 have shown that the Levenberg-Marquardt algorithm provides a more stable solution to the optimization problem. However, switching to the Quasi-Newton method is sometimes helpful to overcome critical points where the Levenberg-Marquardt algorithm terminates early.

3.4.2 Simulated Annealing

While the Levenberg-Marquardt algorithm is an efficient way of detecting the minimum of an objective function that is convex within the parameter space of interest, it cannot assure convergence to the *global* minimum. As a consequence, the parameter set at a local minimum does not have the quality of being a maximum likelihood estimate. The method of Simulated Annealing is a technique to find the (ideally *global*) minimum of the objective function in the presence of many local minima. The basic idea is an analogy with thermodynamics, specifically with the way metals slowly cool and anneal. The analogy is described in *Press et al.* [1992]. For our minimization purposes let's assume that the simulated thermodynamic system changes its configuration from $\zeta(\mathbf{p}_k)$ to $\zeta(\mathbf{p}_{k+1})$ with probability

$$p = \exp(-\Delta\zeta/T) \qquad (20)$$

where $\Delta\zeta=\zeta(\mathbf{p}_{k+1})-\zeta(\mathbf{p}_k)$ and T is a controlling parameter analog to the current temperature during the annealing process. Notice that if $\Delta\zeta$ is negative, the probability p is greater than unity and the step $\Delta\mathbf{p}$ is always accepted as a successful downhill move. However, an uphill move is *sometimes* taken with the probability given by (20). In order to perform Simulated Annealing Minimization one must provide the following elements:

1. Define the range of possible parameter values. Define an initial control parameter T_0; the value of T_0 should be a reasonable fraction of the initial objective function, e.g. $0.1 \cdot \zeta_0$.

2. Generate random perturbations $\Delta\mathbf{p}$ of the parameter vector $\mathbf{p}$. In ITOUGH2, the probability density function of the step size is either Gaussian or uniform; the variances of these distributions decrease during the iteration process.

3. Evaluate the objective function $\zeta(\mathbf{p}_{k+1})$ for the new parameter set $\mathbf{p}_{k+1} = \mathbf{p}_k + \Delta\mathbf{p}$.

4. If the objective function decreases ($\Delta\zeta$ is negative), retain the change and return to step 2. After a sufficient number of perturbations has been accepted, lower T according to the so-called *annealing schedule* which tells after how many perturbations T is updated, and how much the reduction of T will be. ITOUGH2 provides two annealing schedules:

$$T = \alpha^k T_0 \qquad (21a)$$

where α is a user specified constant ($\alpha < 1$, typically 0.9) and k is the number of updates. The second annealing schedule is given by

$$T = T_0 (1 - k/K)^{\beta} \tag{21b}$$

where K is the total number of iterations foreseen; β is a user specified constant.

5. If $\Delta\zeta > 0$, accept the perturbation with probability given by (20).

This scheme of always taking a downhill step and *sometimes* taking an uphill step with probability p depending on T has come to be known as the Metropolis algorithm.

The main drawback of Simulated Annealing is its inefficiency which is due to the randomness of the step $\Delta\mathbf{p}$ which almost always proposes an uphill move. More sophisticated schemes for choosing $\Delta\mathbf{p}$ have been proposed in the literature - they are not considered here. However, ITOUGH2 offers two options which combine the efficiency of the Levenberg-Marquardt algorithm and the ability of Simulated Annealing techniques to overcome local minima. The basic idea of the first option (option A) is the following:

1. Perform Simulated Annealing optimization for a given number of iterations. The aim of this initial step is to detect the convex region containing the global minimum without actually performing the inefficient minimization near the optimum.

2. Switch to the Levenberg-Marquardt algorithm to efficiently detect the global minimum.

The second option (option B) can be used to check whether the solution obtained by the Levenberg-Marquardt algorithm is a global minimum:

1. Perform minimization of the objective function using the Levenberg-Marquardt algorithm.

2. Perform an *a posteriori* error analysis (see Section 3.5). Calculate the covariance matrix of the estimated parameter set; perform an eigenanalysis of this matrix; select the eigenvector associated with the smallest eigenvalue as the search direction for Simulated Annealing minimization.

3. Perform a one-dimensional minimization along this vector in the n-dimensional parameter space using Simulated Annealing.

4. If a lower value of the objective function is detected, return to step 1.

The idea of selecting the line given by the eigenvector associated with the smallest eigenvalue as the search direction is somewhat intuitive. It is suspected that the region containing the global minimum - if not yet detected - is most probably found following this direction. The concept of option B is schematically illustrated for two parameters p_i and p_j in Figure 4. For more details about the Simulated Annealing optimization see Section 4.3.5.2.

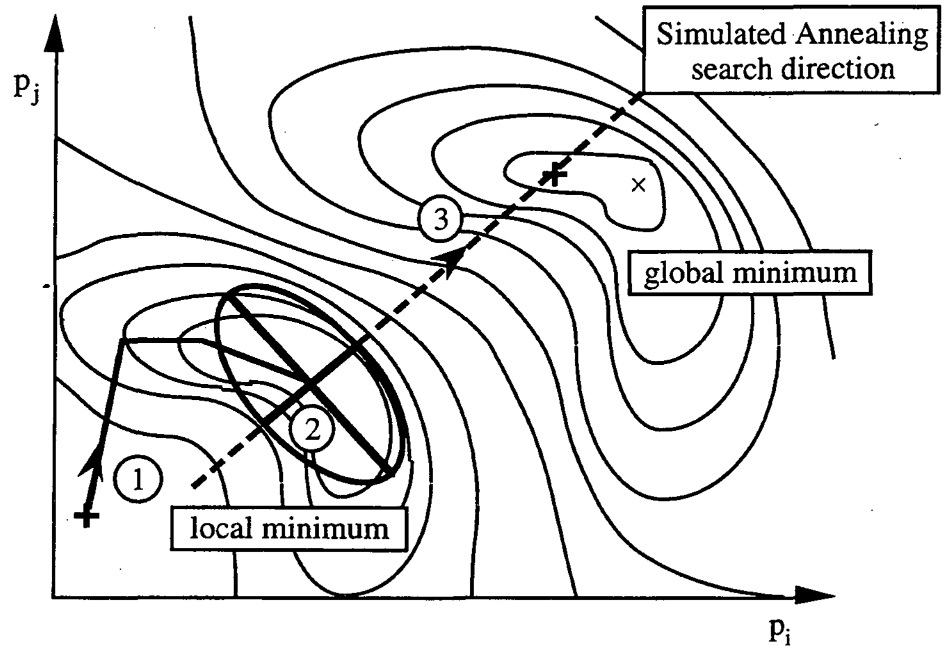


Figure 4: Combination of Levenberg-Marquardt and Simulated Annealing optimization

3.5 Error Analysis

In order to assess the reliability of the parameter estimates, ITOUGH2 provides an *a posteriori* error analysis. First, an estimate of the empirical error variance is obtained:

$$\hat{\sigma}_0^2 = \frac{(\mathbf{z}^* - \hat{\mathbf{z}})^T \mathbf{V}^{-1} (\mathbf{z}^* - \hat{\mathbf{z}})}{m - n} \tag{22}$$

where the test statistic follows an F-distribution where the first degree of freedom is (m-n), and the second degree of freedom is ∞. If the ratio $\hat{\sigma}_0^2/\sigma_0^2$ significantly deviates from one, either matrix **V** or the formulation of the direct problem is erroneous. This so-called Fisher Model Test can only be made if measurement and computation errors are accurately quantified *a priori*. Since the covariance matrix **C** is usually not well known, formula (22) may be used to estimate the error variances $(\hat{\sigma}_0^2)_i$, $i \in \{p,pre,flow,temp,sat\}$ for each of the observation types. ITOUGH2 allows calculation of these quantities simultaneously with the other model parameters (see Section 4.3.5.4).

Once the *a posteriori* error variance is calculated, a first-order approximation of the parameter covariance matrix is obtained as

$$\mathbf{C}_p = \hat{\sigma}_0^2 \cdot (\mathbf{J}^T \mathbf{V}^{-1} \mathbf{J})^{-1} \tag{23}$$

where **J** is the Jacobian at the solution $\hat{\mathbf{p}}$. Matrix $\mathbf{C}_p$ is a measure of the estimation error. The diagonal terms represent the variances of a joint probability density function which describes the variability of parameter p_i taking into account the variability of all the parameters which have been estimated simultaneously. If the estimates are correlated, the uncertainty of one parameter does affect the uncertainty of another parameter. As a consequence, the variances of the estimated parameters may be too optimistic due to the disregarded uncertainty of all the parameters that have been fixed during the formulation of the direct problem, thus being part of the model structure. In comparison, the conditional standard deviation σ_{pi}^* measures the uncertainty of a parameter provided that all the other parameters are exactly known or uncorrelated. The situation is illustrated in Figure 5 for the case of two parameters.

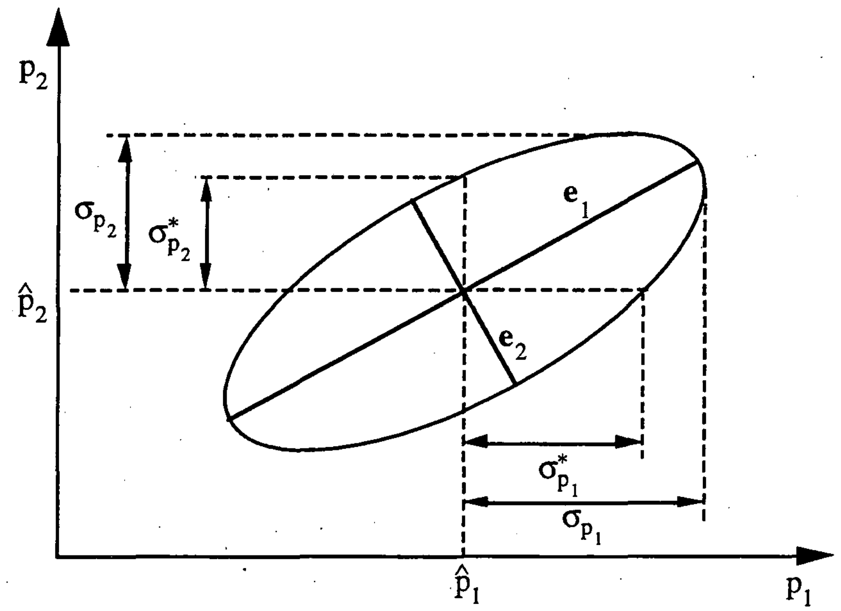


Figure 5: Two-dimensional confidence region

Shape and size of the confidence region can be determined by an eigenanalysis of matrix $\mathbf{C}_p$. The length of the semiaxis $\mathbf{e}_i$ is proportional to the square root of the eigenvalue a_i^2. Therefore, large eigenvalues correspond to linear combinations of parameters that are poorly estimated. The coefficients of the linear combination, the orientation of the semiaxis, are the components of the corresponding eigenvector. The factor of proportionality can be interpreted as a function of the $F_{n,m-n,1-\alpha}$-quantile which determines the probability that the true parameter set $\mathbf{p}$ is within the confidence region. The linearized confidence region of significance level α is defined by those values $\bar{\mathbf{p}}$ for which

$$(\bar{\mathbf{p}} - \hat{\mathbf{p}})^T \mathbf{C}_p^{-1} (\bar{\mathbf{p}} - \hat{\mathbf{p}}) \leq \hat{\sigma}_0^2 \cdot n \cdot F_{n,m-n,1-\alpha} \qquad (24)$$

The scaled condition number is defined as the ratio between the largest and the smallest eigenvalue, divided by the value of the parameter p_i. The scaled condition number is a measure of ill-conditioning in the estimation problem at hand. An inspection of $\mathbf{C}_p$, eigenvalues, eigenvectors, and condition number may give useful information about uncertainty, correlation structure, and conditioning of the parameter estimates in the view of the available data under the given flow conditions.

The linear error analysis outlined above implies that the confidence region at a reasonable level is small enough to approximate the calculated system response $\hat{\mathbf{z}}$ as a linear function of the parameters. It can be shown that (23) is merely a lower bound on the actual estimation covariance matrix if the model response is a nonlinear function of the parameters and the sample size is not large enough. *Carrera* [1984] proposed a method for correcting the covariance matrix so that nonlinear effects can be approximately accounted for. We adapt his basic idea of comparing the actual likelihood function with the results from the linear approximation at discrete points in the parameter space. These test points are preferably located along the axis of the hyperellipsoid:

$$\tilde{\mathbf{p}}_{i\pm} = \hat{\mathbf{p}} \pm \sqrt{n \cdot F_{n,m-n,1-\alpha}} \cdot a_i \cdot \mathbf{u}_i \qquad (i=1..n) \qquad (25)$$

Here, $\tilde{\mathbf{p}}_{i\pm}$ are two test parameter sets on the i-th axis, the direction of which is given by the eigenvector $\mathbf{u}_i$ of the covariance matrix $\hat{\sigma}_0^{-2} \cdot \mathbf{C}_p$. Note that the distance from the optimal parameter set $\hat{\mathbf{p}}$ is selected as a multiple of the corresponding eigenvalue a_i^2 and the quantile of the F-distribution. This means that the correction is tailored to approximate the confidence region on a certain confidence level $1-\alpha$. The eigenvalues a_i^2 which represent the length of the semiaxis are now corrected as follows:

$$a'^2_i = a_i^2 \cdot \hat{\sigma}_0^2 \cdot \left(\frac{A_+ + A_-}{2} \right)_i \qquad (26)$$

with

$$A_{\pm i} = \frac{n \cdot F_{n,m-n,1-\alpha}}{\zeta(\tilde{\mathbf{p}}_{i\pm}) - \zeta(\hat{\mathbf{p}})} \qquad (27)$$

Finally, the new covariance matrix is backcalculated from the eigenvectors $\mathbf{u}_i$ and the updated eigenvalues a'^2_i. The proposed correction requires 2n additional solutions of the direct problem and is thus relatively inexpensive. While the resulting confidence region is ellipsoidal by definition, the differences between $\zeta(\tilde{\mathbf{p}}_+)$ and $\zeta(\tilde{\mathbf{p}}_-)$ provide - as a byproduct of the correction procedure - some insight into the asymmetry of the actual confidence region.

A complete understanding of the error structure, i.e. the actual confidence region, can be obtained by evaluating (8) in the vicinity of the estimated parameter set. The 100(1-α)% confidence region for the true but unknown parameter vector $\mathbf{p}$ contains those values $\bar{\mathbf{p}}$ for which

$$\zeta(\dot{\mathbf{p}}) - \zeta(\hat{\mathbf{p}}) \leq \hat{\sigma}_0^{\,2} \cdot n \cdot F_{n,m-n,1-\alpha} \tag{28}$$

First draw a contour map of the objective function in the vicinity of the optimum parameter set $\hat{\mathbf{p}}$ by using the appropriate ITOUGH2 option. Then, the actual confidence region is bounded by the contour level $\zeta(\hat{\mathbf{p}}) + \hat{\sigma}_0^{\,2} \cdot n \cdot F_{n,m-n,1-\alpha}$. In order to check the validity of the linearity assumption, this contour can be compared to the ellipsoidal confidence region obtained from the linear error analysis outlined above. Obviously, this procedure can only be applied for $n \leq 3$.

After having evaluated the final residuals $r_i = z_i^* - \hat{z}_i$, ITOUGH2 calculates matrix $\hat{\mathbf{C}}$ holding the covariances of the calculated system response:

$$\hat{\mathbf{C}} = \mathbf{J}\,\mathbf{C}_p\,\mathbf{J}^T \tag{29}$$

We define a covariance matrix $\mathbf{C}_r$ as follows:

$$\mathbf{C}_r = \mathbf{C} - \hat{\mathbf{C}} \tag{30}$$

From this, three measures of reliability are calculated [*Baarda*, 1968]. The first, defined as

$$y_i = \frac{\sigma_{r_i}}{\sigma_i} \tag{31}$$

is the so-called *local reliability*, where σ_i is the variance of the i-th observation, i.e. the diagonal element of matrix $\mathbf{C}$, and σ_{ri} is the corresponding diagonal element of matrix $\mathbf{C}_r$. The local reliability realizes values between zero and one. It is a measure of how much the individual measurement is controlled by redundant observations. If y_i is close to zero, even a large error of the corresponding observation cannot be detected (see discussion of Equ. (33) below). A value $y_i = 100\%$ indicates a totally controlled observation. Adding more observation points in the vicinity of this measurement does not improve the reliability of the inverse modeling system and is therefore unnecessary. Note that y_i can be evaluated without actually performing the measurements if the *a priori* covariance matrix is expected to pass the Fisher Model Test. The elements of $\mathbf{C}_r$ then only depend on the number and location of the observation points and their sensitivity with respect to the model parameters. Therefore, they may be used to improve the design of an experiment.

Next, the *normalized residual*:; w_i is evaluated as:

$$w_i = \frac{r_i}{\sigma_{r_i}} \tag{32}$$

w_i is a normally distributed variable with $E[w] = 0$ and $\sigma_w = 1$. Therefore, it is possible to test each observation with the corresponding w_i. If the realization w_i is larger than a predefined $w_{i,max}$, then the corresponding observation is likely to be erroneous and will be marked with a "*" in the ITOUGH2 output file. The quantile $w_{i,max} = u_{1-\alpha}$ is calculated internally given a certain confidence level $(1-\alpha)$. Here, α is the risk to reject an observation even though it is correct.

Finally, ITOUGH2 provides an estimate of the *smallest detectable error*:

$$\nabla r_i = \frac{\sigma_{r_i}}{y_i} \cdot (u_{1-\alpha} + u_{1-\beta}) \tag{33}$$

where β is the risk that an error of size ∇r_i is not detected.

3.6 Model Identification Criteria

As mentioned earlier, maximum likelihood estimation leads to optimum parameters *for a given model structure*. However, this does not imply that the representation of the real system is satisfying at all. If the conceptual model fails to reproduce the salient features of the groundwater system, the calibrated model may not be able to match the observed data as expected (note that our expectation regarding the fit is reflected in the *a priori* covariance matrix of the residuals, **C**). The Fisher Model Test outlined in Section 3.5 is a first indication of whether the model fits the data well enough so that the underlying conceptual model can be accepted. The desire to obtain a good match between observed and predicted system response may tempt the modeler to increase the number of unknown parameters. Unfortunately, increasing the number of parameters results in a decrease of the parameter reliability because the parameters are strongly correlated and the degree of freedom is reduced; the model may become overparameterized. There is an obvious need for objective criteria to rank alternative models with different model structure. *Carrera* [1986] discusses four model identification criteria. They are all based on a number of assumptions regarding the underlying error structure and its asymptotic behavior (for details see *Carrera* [1984]). The four criteria are given with increasing complexity:

$$\mathrm{AIC}(\hat{\mathbf{p}}) = S(\hat{\mathbf{p}}) + 2n \tag{34}$$

$$\mathrm{BIC}(\hat{\mathbf{p}}) = S(\hat{\mathbf{p}}) + n \cdot \ln(m) \tag{35}$$

$$\phi(\hat{\mathbf{p}}) = S(\hat{\mathbf{p}}) + 2n \cdot \ln(\ln(m)) \tag{36}$$

$$d_M(\hat{\mathbf{p}}) = S(\hat{\mathbf{p}}) + n \cdot \ln(m/2\pi) + \ln(|\mathbf{C}_p^{-1}|) \tag{37}$$

Here, S is the log-likelihood criterion (6), m is the number of observations, n is the number of parameters, and $\mathbf{C}_p$ is the covariance matrix of the estimated parameter set $\hat{\mathbf{p}}$. The model with the lowest value should be chosen among a set of alternatives. In all four criteria, the closeness between the true and the modeled system and the number of parameters are the main contributions. Therefore, the simplest model, i.e. the model with the smallest number of parameters, is chosen if a comparable fit can be obtained. Following this principle, overparameterization can be avoided. The most sophisticated criteria (37) contains the parameter sensitivity matrix $\mathbf{C}_p$. Minimizing the determinant of the Fisher information matrix favors the model with high parameter sensitivities and low correlations between parameters.

While model indentification criteria provide an additional element to qualify the appropriateness of the numerical model, it should be emphasized that they do not account for any "soft" information about the groundwater system the modeler might be aware of. Furthermore, if used as a design tool, additional criteria (e.g. the number of boreholes needed, costs, disturance of the system, etc.) should be used for the evaluateion of the overall test performance, the final design selection, and model discrimination.

3.7 Estimation of Prediction Error

In addition to parameter estimation, ITOUGH2 allows estimation of the errors associated with model predictions. Two options are available:

- *First Order Second Moment (FOSM):*

  If the errors of the model parameters are normally distributed with covariance matrix $\mathbf{C_p}$, FOSM error analysis calculates the covariance matrix of the predicted system state as follows:

  $$\mathbf{C_z} = \mathbf{J'}\ \mathbf{C_p}\ \mathbf{J'}^T \tag{38}$$

  Here, $\mathbf{J'}$ is the Jacobian matrix containing the sensitivity coefficients of the predicted system response with respect to the model parameters, $J'_{ij}=\partial z_i/\partial p_j$, $\mathbf{C_p}$ is the covariance matrix of the parameters holding the variability and correlation structure of all parameters considered uncertain, and $\mathbf{C_z}$ is the calculated covariance matrix of the system response. FOSM error analysis implies that the variances in $\mathbf{C_p}$ are small enough so that the propagation of the uncertainty can be approximated by the first-order term J'_{ij}. However, if the variances are large and/or the model is highly nonlinear, the errors of the system response are usually not normally distributed as assumed by FOSM error analysis.

- *Monte Carlo Simulations (MC):*

  The basic idea of the Monte Carlo method is to randomly generate a sufficiently large number of parameter sets, and to calculate the corresponding system responses which can then be statistically evaluated. ITOUGH2 allows generation of uniformly, normally and log-normally distributed realizations of an individual parameter which are then randomly combined into parameter sets. At present, correlations between the parameters cannot be considered. While MC is computationally expensive, the method provides a more realistic estimate of the true probability density function of the system response because nonlinearities of the prediction model are automatically accounted for.

4. ITOUGH2

4.1 Installation and Execution

In this section, some technical comments are made on how to install and run ITOUGH2. The script files for compilation (Figure 6) and execution (Figure 7) are those used on an IBM RS/6000 workstation under UNIX operating system. However, it is believed that the code should run with very minor modifications on any computer. Machine dependent subroutines are provided for IBM and SUN workstations, STARDENT mini-supercomputers, and CRAY supercomputers.

ITOUGH2 is written in standard FORTRAN-77. 64-bit arithmetic is required and a minimum core memory of about 8 MBytes is recommended. The source code consists of five files plus the slightly modified TOUGH2 source files:

it2MAIN.f	:	ITOUGH2 main program
it2INPUT.f	:	ITOUGH2 subroutines to read input file
it2USER.f	:	ITOUGH2 user subroutines
mdep???.f	:	machine dependent subroutines (e.g. ???=IBM)
it2????.f	:	Interface to library ???? (e.g. ????=IMSL)
tough2.f	:	TOUGH2 files: t2cg1.f, t2f.f, eos*i*.f, meshm.f, ma28.f (t2m.f not needed)
itough2.help	:	Contains help text

It is recommended to run ITOUGH2 in a local directory in which all the input files are copied. ITOUGH2 expects the name of the input files and the name of the corresponding directory to be provided in file <itough2.file>. ITOUGH2 uses a number of additional files; the contents of these files are discussed in detail in the next Section. The UNIX script shown in Figure 7 is a raw version of a command file that generates a temporary directory, runs ITOUGH2, and copies the results back to the original working directory.

A break handler is installed which allows termination of ITOUGH2 at any time during the optimization process. Typing the unix command "`kill -2 PID`" causes ITOUGH2 to complete the current iteration, and to perform the error analysis before stopping.

```
#
# Makefile to make itough2_(EOS).out executable
#
# Set the following seven variables according to your needs
#
# EOS : number of Equation Of State module being used
# COM : name of COMputer (for machine dependent subroutines)
#       (IBM,SUN,CRAY,STARdent)
# LIB : name of LIBrary (none=XXXX,IMSL,ULIB)
# LPA : PAth of Library (only if LIB=IMSL)
# FOR : name of FORtran compiler (e.g. xlf, fc, f77)
# CO1 : Compiler Options for single (1) precision compilation
# CO2 : Compiler Options for double (2) precision compilation
#
EOS    = 3
COM    = IBM
LIB    = XXXX
LPA    =
FOR    = xlf
CO1    = -c
CO2    = -c -O -qautodbl=dblpad
#
OBJ    = it2USER.o mdep$(COM).o it2MAIN.o it2INPUT.o t2cg1.o t2f.o \
         meshm.o eos$(EOS).o ma28.o it2$(LIB).o
#
itough2_$(EOS).out : $(OBJ)
        $(FOR) $(OBJ) -o itough2_$(EOS).out $(LPA)
it2MAIN.o     : it2MAIN.f
        $(FOR) $(CO2)  it2MAIN.f
it2INPUT.o    : it2INPUT.f
        $(FOR) $(CO2) it2INPUT.f
it2USER.o     : it2USER.f
        $(FOR) $(CO2) it2USER.f
it2$(LIB).o   : it2$(LIB).f
        $(FOR) $(CO2)  it2$(LIB).f
t2cg1.o       : t2cg1.f
        $(FOR) $(CO2) t2cg1.f
t2f.o         : t2f.f
        $(FOR) $(CO2) t2f.f
eos$(EOS).o   : eos$(EOS).f
        $(FOR) $(CO2) eos$(EOS).f
ma28.o        : ma28.f
        $(FOR) $(CO2) ma28.f
meshm.o       : meshm.f
        $(FOR) $(CO2) meshm.f
mdep$(COM).o : mdep$(COM).f
        $(FOR) $(CO1) mdep$(COM).f
```

Figure 6: UNIX makefile to compile and link ITOUGH2 (Version IBM RS/6000)

```
#! /bin/sh
# Bourne shell script to run itough2 (Fi, March 12, 1993)
#
# Usage:   itough2 inv dir IEOS
#
#          inv  = ITOUGH2 input file
#          dir  = TOUGH2 input file
#          IEOS = Number of EOS module being used
#
# Provide here the directory where the ITOUGH2 code is installed
prog_dir=$HOME/itough2
#
program=$prog_dir/itough2_$3.out
datum=`date`
ori_dir=`pwd`
inv_fil=`echo $1|awk -F. '{ print $1 }'`
dir_fil=`echo $2|awk -F. '{ print $1 }'`
#
# Create temporary directory:
mkdir $HOME/itough2_$$
cd    $HOME/itough2_$$
tmp_dir=`pwd`                                >> $inv_fil.std  2>&1
#
# Write input file names into file itough2.file
echo $1                                       > itough2.file  2>&1
echo $2                                      >> itough2.file  2>&1
echo $ori_dir                                >> itough2.file  2>&1
echo $datum                                  >> itough2.file  2>&1
#
# Copy input files to temporary directory
cp  $ori_dir/$1 .                            >> $inv_fil.std  2>&1
cp  $ori_dir/$2 .                            >> $inv_fil.std  2>&1
#
# Run itough2
$program                                     >> $inv_fil.std  2>&1
#
# Copy output files to original directory
cp  $dir_fil.sav   $ori_dir                  >> $inv_fil.std  2>&1
cp  $dir_fil.out   $ori_dir                  >> $inv_fil.std  2>&1
cat $inv_fil.err                             >> $inv_fil.out  2>&1
rm  $inv_fil.err   $1                        >> $inv_fil.std  2>&1
cp  $inv_fil.*     $ori_dir                  >> $inv_fil.std  2>&1
cat fort.99                                  >> $inv_fil.std  2>&1
cat itough2.ver                              >> $inv_fil.std  2>&1
cat status                                   >> $inv_fil.std  2>&1
cat itough2.err                              >> $inv_fil.std  2>&1
cp $inv_fil.std    $ori_dir
cd ..
#
# Remove temporary directory
rm -r $tmp_dir
echo "    "
echo "+++++++++++++"
echo "+ ITOUGH2 EOS$3 >>>$inv_fil $dir_fil<<< terminated"
echo "+++++++++++++"
```

Figure 7: UNIX script file to run ITOUGH2

4.2 Program Structure and Disk Files

ITOUGH2 is written in a modular form for easy updating and enhancing the capabilities of both the solution of the direct as well as the inverse problem. Figure 8 shows the program architecture in a simplified flow chart.

The program first reads the TOUGH2 input deck which contains the model structure of the direct problem. Additional information is obtained through the ITOUGH2 input file where the parameters to be estimated, and the observations and their error structure are defined. The minimization algorithm is then launched updating the parameter vector **p** and calling TOUGH2 for the calculation of the system response **q**. The original TOUGH2 code is unchanged, except for sharing some common variables with the new module for inverse modeling capabilities. This program architecture allows updating both the direct and the inverse part of the model more or less independently. More details can be found in the source code.

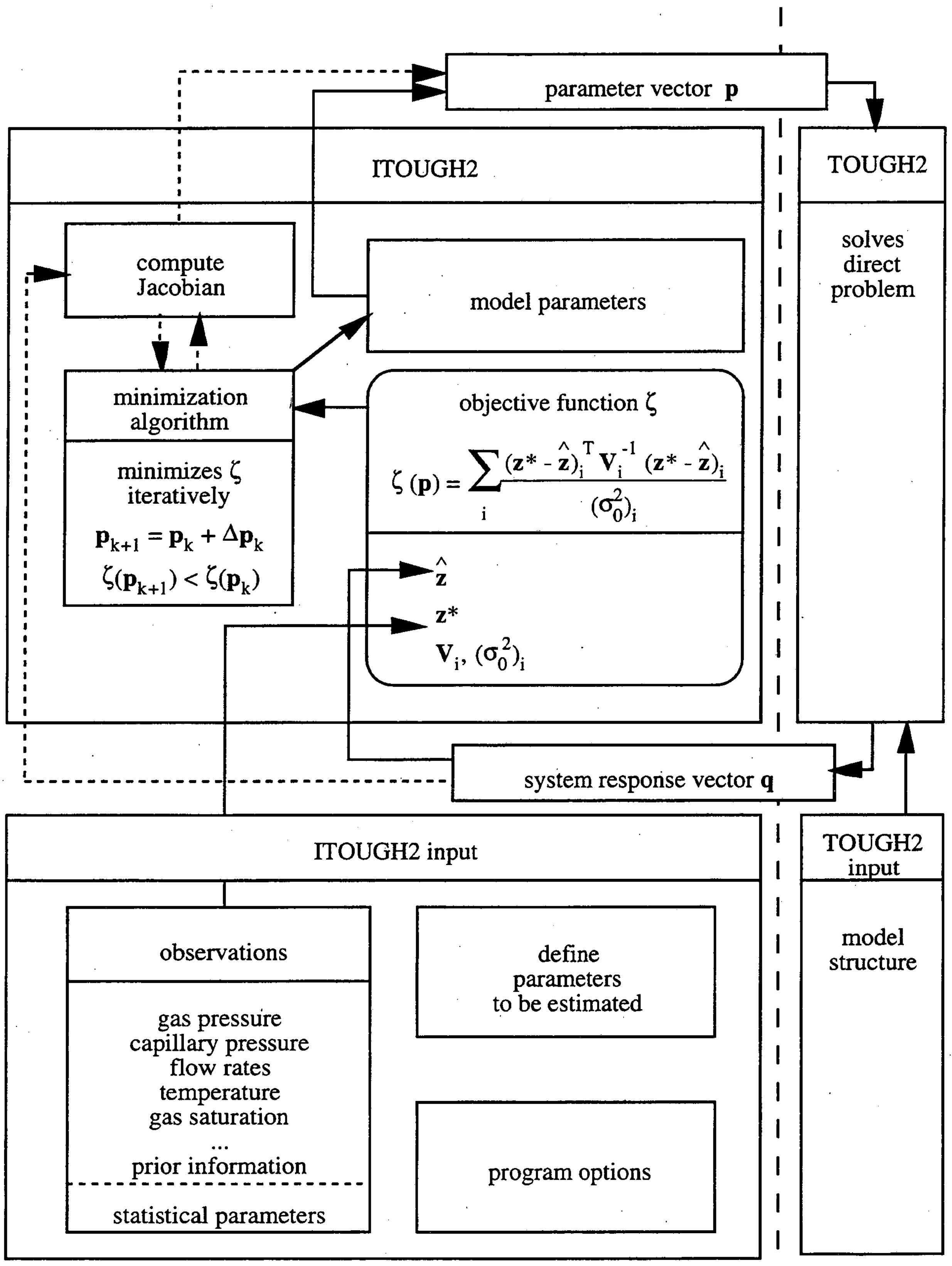


Figure 8: Flow chart of ITOUGH2

ITOUGH2 reads and writes several disk files. The contents of these files are:

Input:

itough2.file:	Contains file name <*invfile*>, <*dirfile*>, and <*working directory*>
<*dirfile*>:	Standard TOUGH2 input deck (see *Pruess* [1987,1991])
<*invfile*>:	ITOUGH2 input file, contains parameter vector, observations and their error structure, program options (see Section 4.3)

Output:

<*dirfile*>.out:	Standard TOUGH2 output file with the system response for the optimum parameter set (see *Pruess* [1987,1991])
<*invfile*>.out:	Contains iteration information, results of error analysis, optimum parameter set, etc. (for an example see Section 5.1.2)
<*invfile*>.par:	Contains optimum parameter set (can be used for restarting)
<*invfile*>.cov:	Contains covariance matrix of the system response (see Equ. 16). (can be used as input to define error structure of observations)
<*invfile*>.xxx:	Contains measured and computed data in appropriate plot format
<*invfile*>_mc.xxx:	Contains plot data of Monte Carlo simulations (only if requested)
<*invfile*>_ch.xxx:	Contains plot data of characteristic curves (only if requested)
<*invfile*>.err:	Contains summary of ITOUGH2 error messages
<*invfile*>.std:	Contains error messages from unix operating system
itough2.ver:	Contains short printout for version control
status:	Displays current status. This file is updated after each iteration and can be used for on-line checking

The extension *.xxx* of the plotfiles depends on the interface being used. By default, ITOUGH2 generates ASCII files for the TECPLOT plotting program (extension *.tec*).

Most error and warning messages are displayed in file <*invfile*>.out for ITOUGH2 errors and <*dirfile*>.out for TOUGH2 errors. If these files are empty, check file <*invfile*>.std for possible messages from the unix operating system. This file also contains some useful information after a successful ITOUGH2 run.

4.3 Preparation of Input File

4.3.1 Basic Concept of Input Language

Execution of ITOUGH2 is controlled using a high-level input language the syntax of which is described in the following sections. The commands are in English (written by a Swiss and therefore probably erroneous) and are hierarchically structured. The command level is identified by a special marker (e.g. ">>" to enter command level two) followed by one or more keywords that triggers a particular action by the program when it is read. The order of the commands within a command level is irrelevant. Some commands are followed by a colon and one or more integers, reals or character strings. Each command level has to be terminated by reversed markers (e.g. "<<" to quit command level two). Any line not containing a command marker or data following a command is considered to be a comment. Sections of the input file may be skipped by prefacing them with the characters /* and following them with the characters */. Lines between these two markers are not interpreted at all. The keyword `HELP` can be added on a command line to print a short tutorial of the corresponding command. Print `>>> LIST` to obtain a list of acceptable commands on the actual command level.

In the detailed description of individual commands, the words being interpreted as an ITOUGH2 command are written in capitals. However, the spelling of the commands in the ITOUGH2 input file is not case-sensitive. The type of a variable following a command is indicated by *italics*. The possible types are: *integer*, *integer list*, *real*, *real list*, *string*, and *string list*. Optional parts of commands are shown in parentheses.

Each line of the input file is processed word by word, the delimiter being one or more spaces. Commands and data are read in free format. The identifiers preceded by a colon have to be written on the same command line they refer to. Long lists of data start one line after the command line. Data lists will be read as long as no reading error occurs.

The open syntax of the input language allows for a flexible organization of the input file. This reference manual will show some examples; however, other arrangements of the input deck may be more appropriate.

The preparation of a valid ITOUGH2 input deck (file <*invfile*>) is described in the following sections.

4.3.2 Main Structure

There are three commands of command level one. They read subcommands which define the model parameters to be estimated, the observations of the system response, and a number of program options and administrative functions. The three commands are:

```
> PARAMETER
> OBSERVATION (or MEASUREMENT)
> COMPUTATION (or OPTION)
```

In the following sections, the available subcommands of these three main commands are discussed in detail. A brief explanation of the purpose of each command is given, as well as definitions of its associated keywords and control parameters.

4.3.3 Definition of Model Parameters

This section describes the subcommands of the first level command `> PARAMETERS` defining the model parameters for which a value has to be determined by inverse modeling. Table 1 summarizes the parameters which can be selected for inverse modeling purposes.

>> Command	TOUGH2 variable	value / log / factor	Parameter
ABSOLUTE	PER(1)-PER(3)	**log** / factor	absolute permeability
CAPILLARY	CP(1) - CP(7)	**value** / log / factor	capillary pressure function
COMPRESS	COM	**log** / factor	compressibility
GENERATION	GX	value / log / **factor**	constant generation rate
INITIAL	DEPU(i) or DEP(i)	value / **factor**	(default) initial conditions
MINC	PAR(i)	**value** / factor	MINC parameters
POROSITY	POR	**value** / log / factor	porosity
RELATIVE	RP(1) - RP(7)	**value** / log / factor	relative permeability functions
SELEC	FE(i)	**value** / log / factor	SELEC parameters
USER	any	**value** / log / factor	user specified parameters

Table 1: Possible model parameters to be estimated

The first column shows the ITOUGH2 commands of level two. They may be accompanied by additional keywords and various subcommands described later in this section. The second column holds the corresponding TOUGH2 variable which will be affected during the optimization procedure. Column three indicates whether the value of the parameter itself, the logarithm of the value, or a scale factor can be estimated; the default option is printed bold.

Most of the parameters are associated with a certain rock type or a set of variables which holds default properties. These options are chosen using one of the following commands of level three:

```
>>> DEFAULT
>>> ROCK (or MATERIAL, or SOURCE): string list (+ integer)
```

The rock type is specified by a five-character string. Blanks should be denoted by underscores (e.g. 'MARL_' for 'MARL '). If more than one name are given, a single parameter value is estimated for all these materials, or, if a factor is to be estimated, all parameter values of the corresponding rock type are multiplied by the estimate. If a '+' is followed by an integer NADD, NADD successive rock types are generated whereby the code number (last two characters) of the last name is incremented by one.

Further information is provided through a number of fourth level subcommands:

```
>>> ANNOTATION: string
```

A 15-character string can be given to describe the parameter. If no parameter annotation is provided by the user, ITOUGH2 generates an annotation which allows easy identification.

Most parameters have to be specified in more detail. The absolute permeability, for example, has three components, PER(i) i=1..3. Furthermore, the user specifies up to seven parameters for the relative permeability and capillary pressure functions (see TOUGH2 variables RPD(i), RP(i), CPD(i), and CP(i), i=1..7). The index *i* (or a list of indices) is selected using the following subcommand:

```
>>>> INDEX (or PARAMETER or VARIABLE): integer list
```

Three keywords are used to determine whether the value, the logarithm of the value, or a multiplication factor is to be estimated. The valid choices and the default options are listed for each of the parameters in Table 1. The keywords are:

```
>>>> VALUE
>>>> LOGARITHM
>>>> FACTOR
```

Example: if the commands `>> INITIAL COND.: 2` and `>>> MATERIAL: ROCK1` are followed by the subcommand `>>>> VALUE`, then the initial gas saturation of all grid blocks with material name ROCK1 is estimated as a single value, whereas subcommand `>>>> FACTOR` causes the estimation of a multiplication factor for the gas saturation of the corresponding grid blocks as specified in the TOUGH2 data block INCON or PARAM.4.

With the next group of keywords, the diagonal elements of matrix $\mathbf{C}_p$ (see Equ. (3)) are assigned. They determine the weight of the prior information about the parameters in relation to the observations of the system response. While the elements of $\mathbf{C}_p$ are in fact variances, ITOUGH2 allows to specify the following quantities alternatively:

```
>>>> VARIANCE:  real
>>>> DEVIATION: real
>>>> WEIGHT:    real
>>>> RELATIVE:  real (%)
>>>> AUTO
```

The real value after the colon is either the variance, the standard deviation, the weight (equivalent to the reciprocal of the standard deviation), or a relative error (in percent or as a fraction), respectively. The last keyword invokes an automatic scaling which simply takes the reciprocal of the value. This option should only be used if applied for all parameters and observations.

In order to make sure that the parameters observe physical or computational restrictions, it is recommended to specify bounds on the parameters, defining an acceptable range of parameter values:

```
>>>> BOUNDS (or RANGE): real real
```

The first real is the lower, the second the upper bound on the parameter. Specifying bounds is useful to prevent TOUGH2 from aborting during the iteration process, e.g. caused by assigning a negative value for porosity. However, the solution of the inverse problem has to be questioned if one or more estimates is fixed on a boundary.

The next two keywords deal with Monte Carlo simulations (see Section 3.7) where the probability density functions of the input parameters have to be specified. There are two options:

```
>>>> NORMAL (or GAUSS)
>>>> UNIFORM
```

Using the first keyword, ITOUGH2 generates a set of normally distributed parameters. The mean of the distribution is the initial guess of the parameter; the variance is specified as described earlier in this section. Uniformly distributed parameters between the lower and upper bound may be generated using the keyword `>>>> UNIFORM`.

While the most frequent parameters are selected by predefined keywords (see Table 1), the user may wish to estimate additional TOUGH2 input parameters. This is easily done by using the following command:

```
>> USER: string
```

The string variable after the command `USER` contains the parameter annotation which is can be used to identify the type of the parameter. The user has to provide the corresponding TOUGH2 variable in subroutine USERPAR (file it2USER.f). Figure 9 shows a FORTRAN listing of subroutine USERPAR. As an example, the tortuosity factor is estimated.

```
***********************************************************************
      SUBROUTINE USERPAR(IUIG,XX,IVLF,IDA,NAMEA,ANNO)
***********************************************************************
* User specified parameters                                           *
* IUIG    = 1 : Provide initial guess (input)                         *
*         = 2 : Update parameter                                      *
* XX      = ITOUGH2 variable = parameter to be estimated              *
*           (output if IUIG=1, input if IUIG=2)                       *
* IVLF    = 1: value (input)                                          *
*         = 2: logarithm                                              *
*         = 3: factor                                                 *
* IDA     = array with parameter IDs (if needed) (input)              *
* NAMEA   = array with material or element names (if needed) (input)  *
* ANNO    = parameter annotation as specified in ITOUGH2 input deck   *
***********************************************************************

      PARAMETER (MDOM=27)
      CHARACTER NAME*5,ANNO*15
      DIMENSION IDA(*),NAMEA(*)

C --- TOUGH2 common blocks
      COMMON/SOLI/COM(MDOM),EXPAN(MDOM),CDRY(MDOM),TORT(MDOM),GK(MDOM)

      CALL GETNMAT(NAMEA(1),NMAT)
      IF (IUIG.EQ.1) THEN
C --- Provide initial guess for variable TORT through TOUGH2 input deck
        XX=TORT(NMAT)
      ELSE
C --- Update TOUGH2 variable TORT at each iteration
        TORT(NMAT)=XX
      ENDIF
      END
```

Figure 9: Subroutine USERPAR: Estimate user specified parameter tortuosity

Finally, an initial guess $\mathbf{p}_0$ for each parameter has to be provided either through the TOUGH2 or the ITOUGH2 input file. The initial guess is the point in the parameter space from which the optimization procedure starts. ITOUGH2 provides several options to define initial guesses. If none of the following commands is used, the value as specified in the TOUGH2 input file is taken as the initial guess. This might be the most elegant way since ITOUGH2 automatically assigns the values to the variables in the parameter vector $\mathbf{p}$. The second possibility is to assign the initial value directly to a parameter using the fourth level command:

```
>>>> GUESS (or PRIOR information): real
```

This command overwrites the initial guess provided by the TOUGH2 input deck.

The third possibility is to use a second level command as follows:

```
>> GUESS (or PRIOR information)
   integer  real
   integer  real
   ...      ...
```

The integer indicates the number of the parameter in vector **p** according to the sequence as the parameters enter the ITOUGH2 input file. The real value is the corresponding initial guess. The correct sequence is also provided on file <*invfile*>.par which may be used to restart the optimization just by copying its contents after command >> `GUESS` or by using the keyword `FILE` on the command line which causes ITOUGH2 to read the file specified after the colon.

```
>> GUESS (or PRIOR information) from FILE: string
```

The second-level command >> `GUESS` overwrites both the fourth level command >>>> `GUESS` and the initial guess provided by the TOUGH2 input file. Usually, the initial parameter vector $\mathbf{p}_0$ is identical with the vector $\mathbf{p}^*$ which holds the prior information about the parameters. Any difference between the optimum parameter estimate $\hat{\mathbf{p}}$ and $\mathbf{p}^*$ will contribute to the objective function, weighted by the inverse of $\mathbf{C}_p$. However, if the prior information about the parameter $\mathbf{p}^*$ is not identical with $\mathbf{p}_0$, then the prior information has to be defined using the fourth-level command, whereas the starting point is defined using the second-level command.

The following examples demonstrate some of the options described in this section.

```
** This is an ITOUGH2 input file example.
It demonstrates how to define the parameters which are
to be estimated by inverse modeling **

The following block causes ITOUGH2 to estimate the second
parameter of the default relative permeability function
(variable RPD(2)).
The initial guess is taken from the TOUGH2 input file.

> model PARAMETERS to be estimated
  >> RELATIVE permeability functions
     >>> DEFAULT
         >>>> VALUE
         >>>> PARAMETER No: 2 (residual gas saturation)
         >>>> standard DEVIATION: 0.1
         >>>> accepted RANGE     : 0.0   0.5
         <<<<

The follwing block causes ITOUGH2 to estimate the logarithm of
the first parameter of the relative permeability function
(variable RP(1)) associated with domain ROCK1 and ROCK2. The
initial guess of the logarithm is 1.5.
No weight is assigned to this prior information.

     >>> MATERIAL name      : ROCK1  ROCK2
         >>>> LOGARITHM
         >>>> PARAMETER No.: 1
         >>>> WEIGHT        : 0.0
         >>>> initial GUESS: 1.5
         >>>> RANGE         : 1.0  3.0
         <<<<
     <<< terminate input of relative permeability functions
```

```
The following block causes ITOUGH2 to estimate a factor
(initial guess is 1.0) with which the initial gas saturation
(provided by TOUGH2 block INCON) of all grid blocks with
material name ROCK1, ROCK2, and ROCK3 are multiplied.

 >> INITIAL distribution of primary variable No.: 2
     >>> MATERIAL names : ROCK1 ROCK2 ROCK3
         >>>> ANNOTATION: INI. GAS SAT.
         >>>> VARIANCE  : 0.5
         >>>> BOUNDS    : 0.01 3.6
         <<<<
     <<<

  >> INITIAL PRESSURE field
     >>> DEFAULT
         >>>> ANNOTATION         : Pstat
         >>>> VALUE
         >>>> standard DEVIATION: 0.1E+05          [Pa]
         >>>> PRIOR information : 2.0E+05          [Pa]
         >>>> RANGE             : 1.0E+05 4.0E+05  [Pa]
         <<<<

/* Beginning of commented block

The following block would cause ITOUGH2 to estimate the
logarithm of the initial pressure in the grid blocks with
material name BOUND.

     >>> ROCK type: BOUND
         >>>> LOGARITHM
         <<<<
     <<<
*/ End of commented block
```

```
The following block provides input for a user specified
parameter. The annotation "TORTUOSITY" is transferred to
subroutine USERPAR where TOUGH2 variable TORT is assigned to
the ITOUGH2 variable XX (see Figure 9).

   >> USER specified parameter: TORTUOSITY
      >>> MATERIAL             : ROCK2
          >>>> WEIGHT          : 0.0 (no prior information)
          >>>> RANGE           : 0.1  1.0
          <<<<
      <<<


The following block provides initial guesses for parameter No.
1 and 3. It overwrites the initial guess of 2 bar defined
earlier for parameter 3.

   >> initial GUESS
      1  0.2
      3  1.5E+05

   << terminates main command PARAMETER


> OBSERVATION (see Section 4.3.4)

> COMPUTATION (see Section 4.3.5)
```

Figure 10: ITOUGH2 input file: block PARAMETER

4.3.4 Definition of Observations

The parameters summarized in Table 1 are estimated based on measurements of the system response at discrete points in space and time (Table 2):

>> Command	TOUGH2 variable	Units	Observation
PRESSURE	P(IELM)	[Pa]	gas pressure
LIQUID PRESSURE	P(IELM)+PAR(NLK2L+14)	[Pa]	liquid pressure
CAP PRESSURE	PAR(NLK2L+14)	[Pa]	capillary pressure
GAS FLOW	FLO(NNP+1)	[kg/sec]	gas flow rate
LIQUID FLOW	FLO(NNP+2)	[kg/sec]	liquid flow rate
FLOW: *iph*	FLO(NNP+*iph*)	[kg/sec]	flow in phase *iph*
TOTAL FLOW	FLO(NNP+*iph*),*iph*=1,NPH	[kg/sec]	total fluid flow rate
TEMPERATURE	T(IELM)	[^{o}C]	temperature
GAS SATURATION	PAR(NLK2L+1)	[-]	gas saturation
LIQUID SATURATION	PAR(NLK2L+2)	[-]	liquid saturation
SATURATION: *iph*	PAR(NLK2L+*iph*)	[-]	sat. of phase *iph*
ENTHALPY	EG(i)	[J/kg]	flowing enthalpy
CONCENTRATION	PAR(NLOC2+ (*iph*-1)*NBK+NB+*ic*)	[-]	concentration X_{iph}^{ic}
MASS	XCMASS(*ic*), XPMASS(*iph*)	[kg]	total mass in place
VOLUME	XCVOLU(*iph*)	[m^3]	total phase volume
USER	any	[?]	user specified

Table 2: Possible observation types for calibration

A first command of level two defines the times at which measured and calculated system responses are compared. ITOUGH2 will automatically cause TOUGH2 to stop and to provide output at these times. The calculated value is then compared with the corresponding measurement, linearly interpolated between two data points. There are three options to flexibly define time points:

```
>> TIMES: integer (UNIT)
   real list

>> TIMES EQUALLY spaced: integer (UNIT)
   real  real

>> TIMES LOGARITHMICALLY spaced: integer (UNIT)
   real  real
```

The integer variable indicates the number of time points that will be read or generated. The time *UNIT* can be specified by one of the following keywords:

```
(UNIT) = (SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, YEAR)
```

The second and third option generate `EQUALLY` or `LOGARITHMICALLY` spaced time points between the two values given on the subsequent line. These three command options may be used simultaneously; time ranges may overlap as shown in the following example:

Example:

```
>> TIMES: 5 (default is SECONDS)
      60.0        90.0       300.0
   43200.0     86400.0    172800.0

>> TIMES: 5 EQUALLY spaced in MINUTES
       5.0        50.0

>> LOGARITHMICALLY spaced TIMES: 10 in HOURS
       1.0        24.0
```

This sequence will generate the following points in time [sec] at which calibration will be performed:

```
      60.0        90.0       300.0       600.0      1200.0
    1800.0      2400.0      3000.0      3600.0      5125.0
    7200.0      7295.0     10380.0     14780.0     21040.0
   29950.0     42640.0     43200.0     60700.0     86400.0
```

The TOUGH2 variable TIMAX has to be greater than the largest ITOUGH2 time.

Each time series of measurements is associated with either a TOUGH2 grid block or connection. Pressure, temperature, and saturation measurements are quantities associated with a single grid block, whereas flow rates are evaluated at interfaces between adjacent elements. The appropriate commands are:

```
>>> ELEMENT (or GRID BLOCK): string list (+ integer)
>>> CONNECTION (or INTERFACE): string list (+ integer)
>>> SINK (or SOURCE): string list (+ integer)
```

where each string variable is the code name of a TOUGH2 grid block or sink/source. More than one element, connection, or sink/source code name can be provided on a command line. If a '+' is followed by an integer NADD, NADD successive names are generated whereby the code number of the last name is incremented by one. If more than one location is given, ITOUGH2 calculates either the sum or the mean value of the state variables:

```
>>>> MEAN (ABSOLUTE)
>>>> SUM (ABSOLUTE)
```

By default, data referring to single grid blocks are averaged, whereas for connections (e.g. flow rates across a boundary) the sum of the absolute values (keyword ABSOLUTE) is calculated (to give the total influx to a drift, for example).

Total mass of volumes in place as well as certain user specified observations may not refer to an element or connection. In these cases, a third level command has to be provided as follows:

```
>>> MODEL (or DUMMY)
```

Three options exist to enter the measured data:

Option 1: The observations are entered as a set of paired data. The first value is time, the second value represents the measured value. Again, the times in the list can be given either in SECONDs (default), MINUTEs, HOURs, DAYs, WEEKs, MONTHs, or YEARs; they do not need to correspond to the times as defined in the preceding block using command >> TIME (*UNIT*).

```
>>>> DATA (UNIT)
     real  real
     real  real
     ...   ...
```

The data can also be read from a file. The command then reads:

```
>>>> DATA FILE: string (UNIT)
```

Option 2: If the data can be represented by a polynomial of degree *NPLOY*,

$$f(t) = \sum_{i=0}^{NPOLY} A_i \cdot t^i \tag{23}$$

where t denotes time, ITOUGH2 provides the following option, where the list of reals holds the *NPOLY+1* coefficients A_i.:

```
>>>> POLYNOMIAL of degree: integer
     real list
```

Option 3: The data may also be provided by a user specified subroutine which returns the measured value as a function of time.

```
>>>> USER
```

Before using this option, a subroutine named USERFUNC has to be written, compiled, and linked to the ITOUGH2 code (see file it2USER.f). The parameters are the following:

```
************************************************************
      SUBROUTINE USERFUNC (IDF,TIME,ANNO,VALUE)
************************************************************
* IDF  : Sequence number of time series (input)            *
* TIME : Time at which value is desired (input)            *
* ANNO : Annotation                     (input)            *
* VALUE: Observed value                 (output)           *
************************************************************
```

Figure 11: Subroutine USERFUNC: User specified data

If the units of the observations differ from the TOUGH2 units (see Table 2), the measurements may be multiplied by a factor. The command is:

```
>>>> FACTOR: real
```

Example:
Given a list of pressure measurements in [bar], the appropriate scaling factor is 1.0E+05 to convert the data to [Pa].

The diagonal elements of covariance matrices $\mathbf{C}_i$ $i \in \{p, pre, flow, temp, sat\}$ may be defined by one of the following subcommands:

```
>>>> VARIANCE:  real
>>>> DEVIATION: real
>>>> WEIGHT:    real
>>>> RELATIVE:  real (%)
>>>> AUTO
```

Note that each observation of a given series has the same weight except when using option RELATIVE where the standard deviation is calculated as a relative error for each individual observation. Individual weights can also be assigned using command >> COVARIANCE (see below). Since measurements of different types have different units, the definition of appropriate weights is of great importance. If the absolute errors are not well known, use either option RELATIVE or AUTO, or estimate the scaling factors $(\hat{\sigma}_0^2)_i$, $i \in \{p, pre, flow, temp, sat\}$ according to the procedure outlined in section 4.3.5.4.

A phase has to be specified for flow rates, saturations, and concentrations; a component has to be specified for concentrations:

```
>>>> PHASE: integer
>>>> COMPONENT: integer
```

Note, that phase and component identification can also be made on the second level command, e.g.:

```
>> WATER CONCENTRATION in GAS phase
>> BRINE CONCENTRATION in phase: 2
>> LIQUID FLOW rate
```

No fourth level command is required in this cases.

A final option defines a time window:

```
>>>> TIME WINDOW: real  real
```

The two reals give start and end times the data shall contribute to the objective function. Only one time window may be defined per data set. By default, all data are taken for calibration.

So far only diagonal elements of the covariance matrix **C** have been defined. This means that the errors are assumed to be uncorrelated. However, ITOUGH2 allows assigning values to each element of the covariance matrix **C** using the following subcommand:

```
>> COVARIANCE (DIAGONAL)
   integer  integer  real
   integer  integer  real
   ...      ...      ...
```

or

```
>> COVARIANCE (DIAGONAL) FILE: string
```

The two integers represent the indices of the appropriate matrix element. The structure of matrix **C** is predefined (see Equ. (3)). The keyword `FILE` makes ITOUGH2 read the file specified after the colon; its contents must have the same format as file <*invfile*>.cov which holds the covariance matrix of the calculated system state according to Equ. (38). If keyword `DIAGONAL` is present, only the diagonal elements are taken.

With this option, not only correlations between measurement errors may be assigned but also individual weighting of certain observations (see example below). However, defining a covariance matrix with non-zero off-diagonal elements increases the computational demand; sometimes, the eigenanalysis of such matrices is not stable, either.

If data are available which are not listed in Table 2, the user may write a subroutine USEROBS which provides the corresponding TOUGH2 variable:

```
>> USER: string
```

The string variable contains an arbitrary annotation; the corresponding observation may refer to a single grid block or a connection or neither (e.g. if total gas mass is measured). Figure 12 shows an example of how to add a new data type (provided through common block YOUR_RESULT) to the observation vector $\hat{\mathbf{z}}(\mathbf{p})$.

```
***********************************************************************
      SUBROUTINE USEROBS(IUSER,IOBSA,GRIDA,NECA,INEC,ANNO,TRESULT)
***********************************************************************
* Subroutine No. 8                                                    *
* Provides TOUGH2 result for user specified data type                 *
* IUSER  : Number of dataset (input)                                  *
* IOBSA  : Array containing user specified IDs (input)                *
* GRIDA  : Array containing grid block names (input)                  *
* NECA   : Array containing index of grid block or connection (index) *
* INEC   : Current pointer in arrays GRIDA and NECA (input)           *
* ANNO   : Annotation (input)                                         *
* TRESULT: Provide corresponding TOUGH2 result (output)               *
***********************************************************************

      CHARACTER*5  GRIDA*5, ANNO*15

      DIMENSION IOBSA(*),NECA(*),GRIDA(*)


C --- TOUGH2 Common blocks!
      COMMON/SECPAR/PAR(1)
      COMMON/NN/NEL,NCON,NOGN,NK,NEQ,NPH,NB,NK1,NEQ1,NBK,NSEC,NFLUX

      NEC=NECA(INEC)
      NLOC2L=(NEC-1)*NSEC*NEQ1+NBK
      IF (ANNO.EQ.'BRINE CONTENT') THEN
         SL=PAR(NLOC2L+1)
         XB=PAR(NLOC2L+NB+1)
         TRESULT=SL*XB
      ENDIF
      END
```

Figure 12: Subroutine USEROBS: User specified observations

The following example demonstrates some of the options for data definition.

```
** This is part of an ITOUGH2 input file. It demonstrates how
   to define the observations used for model calibration **

> model PARAMETERS to be estimated (see Section 4.3.3)

> OBSERVATIONS made on system response
  (e.g. constant head injection test + recovery)

  >> TIMES:  4 in SECONDS
       120.0    240.0   7320.0   7440.0

  >> TIMES: 20 for injection period (LOGARITHMICALLY spaced)
       300.0   7200.0

  >> TIMES: 20 for recovery period  (LOGARITHMICALLY spaced)
      7500.0  14400.0

  >> TIMES:  1 for quasi steady state data point
     40000.0

  >> LIQUID FLOW rate (injection)
     >>> CONNECTION : WELL1  ELM_1  (flow from borehole 1)
         >>>> FACTOR: -1.667E-05    (gr/min --> kg/sec)
         >>>> paired DATA set
                140    24.4
                182    14.0
                272    10.6
                ...     ...
               7175     2.4
               7200     2.3
         >>>> RELATIVE error: 5.0  %
         >>>> time WINDOW   : 0.0  7200.0 (injection only)
         <<<<
     <<<
```

```
>> PRESSURE

   >>> ELEMENT          : WELL1
       >>>> ANNOTATION: BOREHOLE 1
       >>>> FACTOR    : 1.0E+05  (bar --> Pa)
       >>>> paired DATA set  (injection + recovery)
                0      2.38
               30      3.46
              600     11.34
              ...      ...
             6956     16.49
             7202     16.66     (shut-in)
             7260     15.32
              ...      ...
            14132     3.54
            14410     3.48
            40000     2.98
       >>>> standard DEVIATION: 0.05 [bar]
       <<<<

   >>> ELEMENT                : INT21 INT22 INT23
       >>>> ANNOTATION        : BOREHOLE 2
       >>>> take AVERAGE of pressures in all intervals
       >>>> POLYNOM of order: 3
            20159.0
               -0.876
                0.00342
               -3.34E-05
       >>>> time WINDOW    : 7200.0 40000.0 (recovery only)
       >>>> RELATIVE error: 0.01 [-]
       <<<<
   <<<
```

```
The following block defines a user specified observation. The
annotation "BRINE CONTENT" is transferred to subroutine USEROBS.
The average brine content (sum of brine mass fraction and liquid
saturation, see Figure 12) is calculated in the region given by
elements ELM01 through ELM20.

  >> USER: BRINE CONTENT
     >>> ELEMENTS: ELM_1  +19
         >>>> AVERAGE brine content in elements 1 through 20
         >>>> DATA [HOURS]
               0.0  0.00
               1.0  0.10
               ...  ...
              10.0  0.90
              12.0  0.95
         >>>> VARIANCE: 0.04
         <<<<
     <<<

  >> COVARIANCE (increase weight of steady state values)
     138   138  2.5E+06
     139   139  2.5E+06

The variances of matrix elements 138 and 139 which correspond to
the last pressure data points in borehole 1 and borehole 2,
respectively, are reduced to increase their relative weight.

  <<

> COMPUTATION (see Section 4.3.5)
```

Figure 13: ITOUGH2 input file: block OBSERVATION

4.3.5 Definition of Program Options

The final command of level one, `> COMPUTATION`, deals with various program options and administrative functions. The following subcommands are available:

```
>> TOLERANCE (or CONVERGENCE, or STOP)
>> OPTION
>> JACOBIAN
>> ERROR
>> OUTPUT
```

4.3.5.1 Convergence, Tolerance, and Stopping Criteria

A number of subcommands deal with convergence, tolerance, and stopping criteria. The parent command is:

```
>> TOLERANCE (or CONVERGENCE, or STOP)
```

ITOUGH2 terminates if one of the following criteria is met:

```
>>> Maximum number of TOUGH2 simulations: integer (-1)
```

ITOUGH2 will terminate if the number of TOUGH2 simulations exceeds the maximum number. Recall, that the direct problem is solved many times mainly to calculate the Jacobian. After termination, the direct problem is solved once more for the optimum parameter set. If `-1` is present on the command line, no final TOUGH2 run will be performed. If the maximum number of TOUGH2 calls is equal to one, the direct problem is solved once without performing any optimization or error analysis. It is strongly recommended to check the solution of the direct problem before switching to inverse modeling. If Monte Carlo simulations are required, the maximum number of TOUGH2 calls determines the number of realizations being generated.

```
>>> Maximum number of ITERATIONS: integer
```

This command reads the maximum number of Levenberg-Marquardt or Quasi-Newton iterations, respectively. At least one iteration is required to perform a complete error analysis.

```
>>> Factor to scale TOLERANCE measures: real
```

The default tolerance measures (scaled gradient tolerance, scaled step tolerance, relative function tolerance, absolute function tolerance, false convergence tolerance) can be multiplied by a user specified factor.

```
>>> Maximum STEP size: real
```

The maximum allowable step size per iteration may be limited. The default value is infinite. Limiting the maximum step size is useful especially when choosing the Quasi-Newton algorithm. Finally,

```
>>> MUE: real
>>> NUE: real
```

reads the Levenberg parameter μ (see Equ. (17), default: 0.001) and the Marquardt parameter ν (default: 10.0) which reduces μ after successful iterations.

ITOUGH2 always stops if a serious error or a warning message has been detected in the input file. It ignores warnings if the following command is given:

```
>>> ignore WARNINGS
```

Command

```
>>> stop after INPUT
```

causes ITOUGH2 to only read and check input without performing any optimization.

4.3.5.2 Program Options

Several program options may be selected. The parent command is:

```
>> OPTION
```

As outlined in Chapter 3, the following minimization algorithms are available:

```
>>>  QUASI-NEWTON
>>>  LEVENBERG-MARQUARDT
```

The Levenberg-Marquardt algorithm is the default method.

Various options of Simulated Annealing minimization as discussed in Section 3.4.2 are described below.

```
>>> ANNEAL (ONLY,BEFORE,SMALL,LARGE,ANY,AFTER)
```

If the keyword `BEFORE` is found on the command line, Simulated Annealing is followed by either the Levenberg-Marquardt or the Quasi-Newton optimization method to detect the minimum of the convex region in the parameter space. The latter will start at the best parameter set obtained by Simulated Annealing. This sequence is referred to as option A.

Keyword `AFTER` invokes the second option (option B) outlined in Section 3.4.2. After completion of the standard minimization procedure, a one-dimensional Simulated Annealing minimization follows, searching for additional minima along the eigenvectors of the covariance matrix (either the `SMALL`est, the smallest and `LARGE`est, or `ANY` eigenvector). Instead of performing Simulated Annealing along these eigenvectors, the user may choose a simple >>> `LINESEARCH` algorithm.

Simulated Annealing requires to specify the following parameters:

```
>>>> ITERATION: integer
```

In terms of the analogy with thermodynamics, one iteration refers to one "temperature step" during the annealing process; the parameter therefore specifies how many times the temperature will be reduced according to the annealing schedule. The number of random steps tried at any temperature is specified by:

```
>>>> STEP: integer
```

By default, the number of steps per iteration is set to ten times the number of parameters to be estimated. The initial temperature can be specified as follows:

```
>>>> TEMPERATURE: (-)real
```

Recall that the term "temperature" here refers to the annealing analogy and thus has a different meaning than the temperature of the physical system that is modeled using TOUGH2. The higher the initial temperature, the more likely the algorithm is to accept an uphill move. If a negative value is given, the initial temperature is internally calculated as the corresponding fraction of the initial objective function. By default, this value is -0.1; thus the initial temperature is 10% of the initial value of the objective function. Finally, the annealing schedule has to be defined:

```
>>>> SCHEDULE: (-)real
```

A positive value indicates that parameter α is given and equation (21a) will be used. A negative value is interpreted as parameter β of equation (21b). By default, ITOUGH2 uses β=1.0. This completes the description of parameters needed for Simulated Annealing optimization.

ITOUGH2 allows evaluation of the value of the objective function for parameter sets which are internally generated. This option might be useful for drawing contour plots of the objective function (see remarks in Section 3.5). The command is:

```
>>> evaluate OBJECTIVE function: integer (integer (integer))
```

A regular grid is generated over the parameter space, bounded by the values as specified by the command `>>>> BOUND` (see Section 4.3.3). The parameter space is then subdivided into n_i points where n_i is the integer value following the colon.

Example:

```
>>> evaluate OBJECTIVE function at: 10 15 locations
```

If two parameters are specified, this subcommands generates a grid with 10×15 points, the first parameter being subdivided into 9, the second parameter into 14 intervals between the appropriate boundaries. TOUGH2 is called 150 times, the value of the objective function is evaluated and printed. If three parameters are given, ITOUGH2 generates a 10×15×15 grid.

The following command causes ITOUGH2 to solve the direct problem without performing any optimization or error analysis:

```
>>> DIRECT problem
```

This is equivalent to specifying the maximum number of TOUGH2 simulations as 1 (see Section 4.3.5.1).

ITOUGH2 allows modification of the default quadratic objective function in order to reduce the impact of large residuals. In addition to the standard least squares optimization, two robust estimators and the L_1-estimator have been introduced in Section 3.3. The default option is the standard quadratic objective function (see Equations 8, 10 and 11) of the nonlinear least-squares formulation:

```
>>> LEAST-SQUARES
```

The impact of large residuals is slightly reduced using the second option (see Equation 14):

```
>>> ROBUST ESTIMATOR 1 (or QUADRATIC-LINEAR): real
```

Similarly, a constant contribution of large residuals to the objective function (see Equation 15) may be defined as follows:

```
>>> ROBUST ESTIMATOR 2 (or QUADRATIC-CONSTANT): real
```

Finally, the L1-estimator (see Equation 12) can also be chosen:

```
>>> L1-ESTIMATOR (or LINEAR)
```

The four different types of objective functions are sketched in Figure 2 for k=1.0. Recall that the minimization algorithm and the error analysis is designed for the default quadratic objective function. The options discussed above may improve the convergence rate if the computed parameter set is far away from the optimum thus leading to large residuals, or if the data exhibit outliers.

In general, ITOUGH2 is used to calibrate time dependent data. Observed and calculated state variables are compared at predefined points in time. It is an essential requirement that the TOUGH2 simulation reaches the last point in time defined by the `>> TIMES` command. If TOUGH2 stops prematurely due to a convergence error, a warning message is printed and optimization eventually terminates. However, one might wish to fit a single observation made under steady-state flow conditions. It is usually difficult to estimate the time at which steady-state is reached. Furthermore, TOUGH2 stops automatically when the change of all primary variables is zero. This may happen at different times, depending on the actual parameter set which is updated during the optimization. In order to address the problem of steady-state data fitting, use the following procedure:

(1) Provide two data points in time, the second point largely exceeding the assumed steady-state time. For example, if a steady-state pressure of 1.5 bar is observed, provide two data points as follows (subcommand of `> OBSERVATION`, `>> PRESSURE`, `>>> ELEMENT`):

```
>>>> DATA [YEAR]
     0.0000   1.5E5   [Pa]
     1001.0   1.5E5
```

(2) Define one point in time for data fitting which largely exceeds the assumed steady-state time (subcommand of `> OBSERVATION`):

```
>> TIME [YEAR]: 1
   1000.0
```

(3) Give the following keyword as a subcommand of `> COMPUTATION`, `>> OPTION`:

```
>>> STEADY-STATE
```

ITOUGH2 now waits until a convergence failure occurs (usually 10 consecutive time steps converging on ITER = 1). Flow conditions are assumed to be steady-state, and the computed output is taken and compared to the measurement. The steady-state time is printed for each TOUGH2 simulation.

Transient data may precede a late time the steady-state data point.

4.3.5.3 Parameters for Computing Jacobian

The Jacobian matrix (18) is calculated by means of a finite difference approximation. The parent command for specifying some parameters for computing the Jacobian is:

```
>> JACOBIAN
```

First, the increment factor for numerically computing derivatives (see factor α in Equ. 19) is assigned as follows:

```
>>> FACTOR: real (%)
```

The factor is given either as a fraction or in percent. The default value α=0.01 has to be increased if the system response is not very sensitive with respect to parameter perturbations.

The finite difference scheme is selected using one of the following subcommands:

```
>>> FORWARD   (: integer)
>>> CENTERED
```

If the first command is chosen, ITOUGH2 calculates the Jacobian as a forward finite difference approximation according to Equ. (18a) with (n+1) TOUGH2 calls per evaluation, where n is the number of parameters to be estimated. Centered finite differences according to Equ. (18b) are selected by the second command. This requires (2n+1) function evaluations. If the command line `>>> FORWARD` contains a colon followed by an integer *iswitch*, ITOUGH2 switches from forward to centered finite differences after *iswitch* iterations.

4.3.5.4 Error Analysis

ITOUGH2 provides a detailed *a posteriori* error analysis as well as some features to estimate prediction errors. The parent command for these options is:

```
>> ERROR
```

ITOUGH2 calculates the estimated error variance $\hat{\sigma}_0^2$ (22). The user has to decide whether the covariance matrix of the parameters (23) and the covariance matrix of the calculated system response (29) are computed based on the *a priori* variance σ_0^2 or the *a posteriori* variance $\hat{\sigma}_0^2$ (default). The selection is done using one of the following subcommands:

```
>>> a PRIORI
>>> a POSTERIORI
>>> FISHER model test
```

If the last command is chosen, the above mentioned selection is automatically made based on the Fisher Model Test (see Section 3.5).

The measures of reliability y_i and ∇r_i as well as the Fisher Model Test require specifying a confidence level $(1\text{-}\alpha)$ and $(1\text{-}\beta)$. The two risks are given either as a fraction or in percent:

```
>>> ALPHA: real (%)
>>> BETA : real (%)
```

The default values are α=5% and β=5%.

As mentioned in Section 3.5, the error variances of each observation type $(\hat{\sigma}_0^2)_i$ $i \in \{p,pre,flow,temp,sat,user\}$ can be estimated iteratively using the following command:

```
>>> estimate SIGMA (or LAMBDA): (-)integer
```

The integer value indicates after how many iterations the error variances are to be recalculated according to (22). We propose an update of $(\hat{\sigma}_0^2)_i$ after 3 to 5 iterations. If a negative number is given, the factor $(\hat{\sigma}_0^2)_p$, which scales the prior information about the parameters, is excluded from the procedure. The use of this option is recommended if the mutual weighting of data of different types is not well known. However, when updating $(\hat{\sigma}_0^2)_i$, the Fisher Model Test is not applicable since no prior estimate of the error structure exists anymore.

The linearity assumption of the error analysis can be checked and a corrected covariance matrix can be calculated following the procedure outlined in Section 3.5. The ITOUGH2 command is the following:

```
>>> test LINEARITY assumption: real (%)
```

The real value is either the increment $k = \sqrt{n \cdot F_{n,m-n,1-\alpha}}$, or - if "%" is present on the command line - the corresponding confidence level $(1\text{-}\alpha)$ in which case the increment k is calculated internally. It should be realized that for a given confidence level the value of k increases dramatically with increasing number of parameters. For example, if 5 parameters are estimated based on 100 observation points, the factor of proportionality to obtain the 95%-confidence region is given by $F_{5,100,0.95} = 2.305 \rightarrow k = \sqrt{n \cdot F} = 3.39$).

Another possibility to improve the accuracy of $\mathbf{C_p}$ is to compute the full Hessian matrix instead of its approximation given by $\mathbf{J^T C J}$. This can be done by

```
>>>  HESSIAN
```

Since the Hessian is computed by means of finite differences, its evaluation requires solving the direct problem as many as 2n+n(n-1)/2 times, n being the number of parameters. Furthermore, it cannot be assured that the Hessian is a positive definite matrix in which case ITOUGH2 automatically uses its approximation which is positive definite by definition.

There are two options to study the uncertainty of model predictions using ITOUGH2: (1) First Order Second Moment (FOSM) error analysis, and (2) Monte Carlo simulations. The two procedures are discussed in Section 3.7. The ITOUGH2 command for the first method is:

```
>>> FOSM (CORRELATION)
    integer  integer  real
    integer  integer  real
    ...      ...      ...
```

The two integers represent the indices of the covariance matrix of the uncertain parameters. The real values are either the variances (diagonal elements) or the covariances (off-diagonal elements). If keyword CORRELATION is present on the command line, off-diagonal elements are interpreted as a correlation factor r_{ij} ($-1 < r_{ij} < 1$). The diagonal elements overwrite the variances specified earlier (see Section 4.3.3). A second possibility is the following:

```
>>> FOSM (CORRELATION) MATRIX: integer
    real list
    real list
    ...
```

If keyword `MATRIX` is present on the command line, the dimension of the covariance matrix is expected after the colon. The lower triangle of the matrix is then given as shown in the following example:

Example:

```
>>> FOSM error analysis, read CORRELATION MATRIX of dim.: 3
    0.1324
    0.45      1.7245
    0.03     -0.78      0.0098
```

The diagonal holds the variances of the three uncertain parameters. Correlation coefficients are specified by the elements in the lower triangle. ITOUGH2 backcalculates the corresponding covariances from the correlation coefficients.

Monte Carlo simulation is invoked by the following command:

```
>>> MONTE CARLO (SEED: integer) (CLASS: integer) (GENERATE)
```

ITOUGH2 will generate as many parameter sets as previously defined using the command `>>> maximum number of TOUGH2 simulations: integer`. The probability density functions can be chosen individually for each parameter by the keywords `>>>> NORMAL`, `>>>> GAUSS`, in combination with `>>>> LOGARITHM` or `>>>> VALUE` (see Section 4.3.3). Parameter values will be generated between the boundaries specified by the `>>>> RANGE` command (see Section 4.3.3). The initial guess is the mean for normal distributed parameters. The parameter values are calculated by means of a random number generator. The seed number and the number of classes the interval is subdivided when drawing histograms can be specified by the user. If keyword `GENERATE` is present on the command line, ITOUGH2 generates and prints the parameter sets without actually performing the Monte Carlo simulations. The user may then check whether the generated probability density function is consistent with the theoretical one before running the large number of simulations.

4.3.5.5 Printout Options

The ITOUGH2 output file contains a large amount of information. First, the input is checked and reprinted for error tracking, if necessary. Convergence information will be provided during the iteration procedure. The Jacobian matrix, sensitivity coefficients, an approximation of the Hessian matrix, covariance and correlation matrix, eigenvalues, eigenvectors and condition number, final residuals and their error measures, and model structure criteria will be printed by default for the optimum parameter set. In this section, subcommands for additional output are discussed. The parent command is:

```
>> OUTPUT
```

The Jacobian matrix and the residuals may be printed after each iteration using the following subcommands:

```
>>> print JACOBIAN
>>> print RESIDUALS
```

The value of the objective function may be printed after each TOUGH2 simulation (and not only after completion of an iteration):

```
>>> print OBJECTIVE function
```

The subcommand

```
>>> NO FINAL
```

prevents TOUGH2 from running the final TOUGH2 simulation with the optimum parameter set. The subcommand

```
>>> VERSION
```

causes ITOUGH2 to print a one-line informative message, identifying the program unit, its version number and date, and the function of the program unit. When making code modifications, these version messages should be appropriately updated to maintain a traceable record of source code developments.

The following three commands give a list of ITOUGH2 subroutines, a list oc references, and a complete command index, respectively:

```
>>> SUBROUTINES
>>> REFERENCES
>>> COMMAND INDEX
```

The last set of commands deals with plotfile contents and format. By default, the plotfile <*invfile*>. xxx contains a table with the measured data, the simulation results for the first TOUGH2 run with the initial parameter set, and the simulation results for the optimum parameter set. ITOUGH2 may print additional simulation results for intermediate iterations:

```
>>> PLOTTING after: integer  iterations
```

If the user wishes a plot of the relative permeability and capillary pressure functions, type:

```
>>> CHARACTERISTIC curves
```

and a new plotfile will be created containing the characteristic curves for all material types specified in the TOUGH2 input deck.

All plotfiles have to be postprocessed by an external plotting package (e.g. PLOPO, TECPLOT, AVS, etc.). The appropriate format of the plotfile can be chosen as follows:

```
>>> PLOTFILE (or FORMAT): string
```

The string variable contains the name of the plotting utility being used to display ITOUGH2 output data. The default format follows the conventions of PLOPO, a plotting program written by U. Kuhlmann (VAW/ETHZ). For any other plot program, ITOUGH2 internally reformats the plot files. An appropriate extension .xxx is added to the file names in order to identify the corresponding plot program (e.g. <*invfile*>.tec for TECPLOT data files). For a list of available interfaces type `>>> FORMAT LIST`. In order to implement new interfaces, add the name of the plot program in subroutine `INPRINT` and perform the corresponding reformatting in subroutine `PLOTIF` and `REFORMAT`.

Finally, the time unit for the output can be selected:

```
>>> (SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, or YEAR)
```

The use of the options discussed in this Section is demonstrated in the sample problem (see Sections 5.1 and 5.2).

5. Sample Problem

A tutorial sample problem is presented in this chapter demonstrating the parameter estimation and error prediction capabilities of the ITOUGH2 code. A synthetic laboratory experiment was chosen to illustrate user options, preparation of input files, and organization of the output. Only marginal efforts are made to interpret the inverse modeling results (more details can be found in *Finsterle* [1993]). First, two parameters of the relative permeability and capillary pressure functions are estimated based on capillary pressure and liquid flow rate measurements (Section 5.1). Subsequently, the impact of parameter uncertainties on model predictions is studied by means of FOSM and Monte Carlo error analysis (Section 5.2).

5.1 Parameter Estimation

5.1.1 The Direct Problem

In order to assess the methodology outlined in Chapter 1, a synthetic experiment under two-phase flow conditions was performed on a computer.

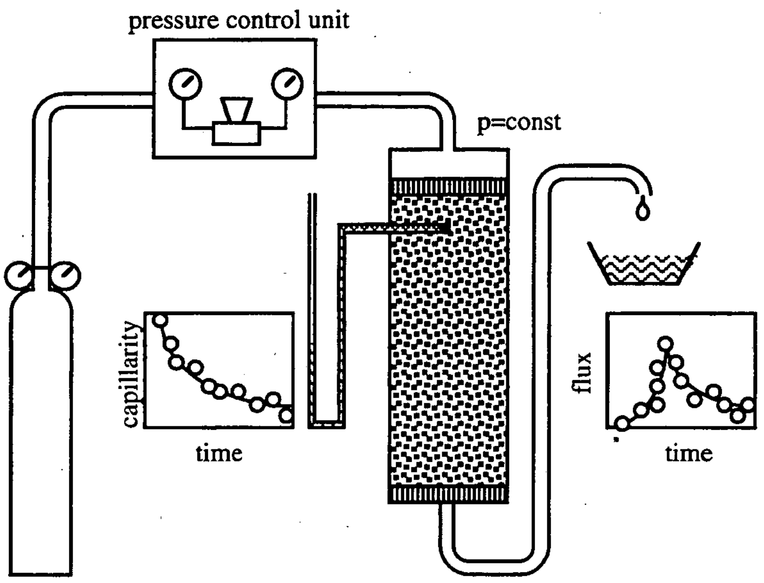


Figure 14: Experimental set-up for synthetic gas injection test

Figure 14 shows the design of a possible laboratory experiment. Gas is injected at a constant pressure of 6 bars into an initially liquid saturated porous medium. In the upper part of the column, a tensiometer is installed that records the capillary pressure as a function of time (Figure 15a). A very precise balance measures the amount of water leaving the bottom of the column. The increasing mass flow rate observed during the first period of the experiment (see Figure 15b) reflects the growing pressure gradient due to the gas-liquid front approaching the lower end of the column. Once the gas has reached the boundary, there is a sharp drop of the liquid flow rate due to reduced relative permeability.

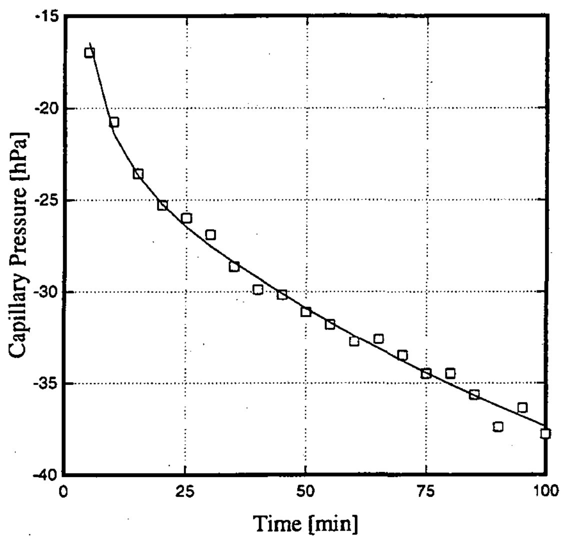


Fig. 15a: Capillary pressure

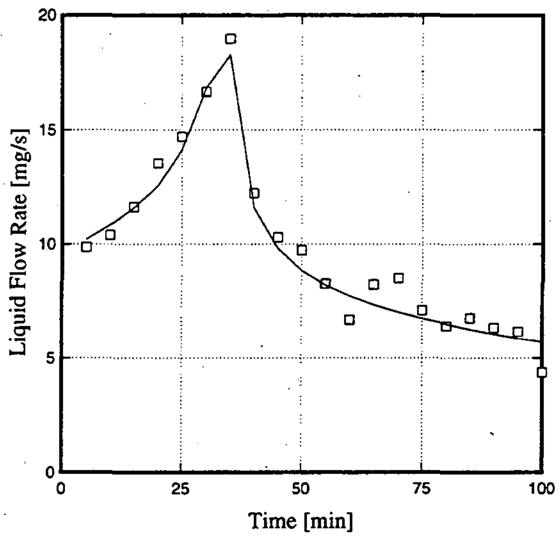


Fig. 15b: Mass flow rate

In order to generate hypothetical data sets, the direct problem was first solved (solid line). Then, the resulting capillary pressures and mass flow rates were corrupted by adding an Gaussian error term (symbols). These data represent the measured system response.

Part of the TOUGH2 input deck for simulating the synthetic experiment is shown below:

```
Synthetic experiment to demonstrate parameter estimation using ITOUGH2
ROCKS (INJEC=pressure control unit, MATRI=rock sample, ATMOS=outlet)
INJEC    2  .250E+04  .1000000  .100E-14  .100E-14  .100E-14  .180E+01  .100E+05
 1.000E-09 0.000E+00  .180E+01 0.000E+00
    5
    1     0.000E+00 0.000E+00  .100E+01
MATRI    2  .250E+04  .1000000  .100E-14  .100E-14  .100E-14  .180E+01  .800E+03
 1.000E-09 0.000E+00  .180E+01 0.000E+00
   11     0.300E+00 0.000E+00
   11     2.500E+00 5.000E-02 1.000E+00
ATMOS    2  .250E+04  .1000000  .100E-14  .100E-14  .100E-14  .180E+01  .100E+05
 1.000E-09 0.000E+00  .180E+01 0.000E+00
    5
    1     0.000E+00 0.000E+00  .100E+01

PARAM
 8 2 300     300100000100020010410101000   .000E-00     1.800
 0.000E+00 1.000E+05       -1. 0.000E+00            9.810     4.000     1.000
  .110E+01

  .1000000000000E+06  .2000000000000E+02 0.0000000000000E+00
ELEME
INJ 1          INJEC 0.100E+50
ELM 0   50    1MATRI 0.100E-03
BOT 1          ATMOS 0.100E+50

CONNE
INJ 1ELM 0                    1 .1000E-10 .5000E-02 .0100E+00    1.0000
ELM 0ELM 1   49    1    1    1 .5000E-02 .5000E-02 .0100E+00    1.0000
ELM50BOT 1                    1 .5000E-02 .1000E-10 .0100E+00    1.0000

GENER

INCON (Result of steady state run to provide static pressure profile)
INJ 1            .10000000E+00
  .6000000000000E+06  .2000000000000E+02 1.0000000000000E+00
ELM 0            .10000000E+00
  .1000489677284E+06  .2000000000000E+02  .0000000000000E+00
ELM 1            .10000000E+00
  .1001469031882E+06  .2000000000000E+02  .0000000000000E+00
ELM 2            .10000000E+00
  .1002448386524E+06  .2000000000000E+02  .0000000000000E+00
.....            .............
  ..................  ..................  ..................
ELM50            .10000000E+00
  .1049457460699E+06  .2000000000000E+02  .0000000000000E+00
BOT 1            .10000000E+00
  .1050000000000E+06  .2000000000000E+02  .0000000000000E+00

START
ENDCY
```

Figure 16: Direct problem: TOUGH2 input deck

5.1.2 The Inverse Problem

Two model parameters - the van Genuchten parameter *n* (pore size distribution index) and $1/\alpha$ (air entry pressure) - will be estimated simultaneously. A new relative permeability and capillary pressure function was coded in TOUGH2 (IRP=ICP=11) in order to have consistent parametric models for the characteristic curves (see *Luckner et al.*[1989]). The ITOUGH2 input file is shown below:

```
This is an ITOUGH2 input file for estimating two parameters of van
Genuchten's relative permeability and capillary pressure functions based
on capillary pressure and liquid flow rate measurements made on the
system response of a synthetic laboratory experiment


  > PARAMETER (see Section 4.3.3)

    >> van Genuchten's CAPILLARY pressure function
       >>> ROCK TYPE                   : MATR1
           >>>> ANNOTATION             : PORE SIZE INDEX n [-]
           >>>> PARAMETER              : 1
           >>>> estimate VALUE
           >>>> standard DEVIATION : 0.25
           >>>> BOUNDS                 : 2.0  3.0
           >>>> PRIOR information      : 2.35
           <<<<
       <<<

    >> van Genuchten's CAPILLARY pressure function
       >>> ROCK TYPE                   : MATR1
           >>>> ANNOTATION             : AIR ENTRY PRES. 1/alpha [bar]
           >>>> PARAMETER              : 2
           >>>> estimate VALUE
           >>>> VARIANCE               : 0.01
           >>>> GUESS                  : 4.00E-02
           <<<<
       <<<


/*  (begin of commented section)

    >> PRIOR information (these are the true values)
    1  2.50
    2  5.00E-02

*/  (end of commented section)


    << (terminate parameter definition)
```

```
> OBSERVATIONS (see Section 4.3.4)

 >> Calibrate at: 20 EQUALLY spaced TIMEs between
    300.0     6000.0 [sec]
 >> CAPILLARY PRESSURE measurements by: TENSIOMETER
    >>> ELEMENT: ELM_5 (= 5 cm below injection)
        >>>> paired DATAset (TOUGH2 results + noise)
        >>>> multiply measurements by FACTOR: 100.0 to obtain [Pa]
        >>>> paired DATAset, time is in MINUTES
                5.0  -0.1698331055E+02 measurements in [hPa]
               10.0  -0.2075763428E+02
               15.0  -0.2357142822E+02
               20.0  -0.2529052490E+02
               30.0  -0.2688769531E+02
               35.0  -0.2862364258E+02
               40.0  -0.2987951172E+02
               45.0  -0.3017028564E+02
               50.0  -0.3111270996E+02
               55.0  -0.3178452455E+02
               60.0  -0.3274365479E+02
               70.0  -0.3348591064E+02
               80.0  -0.3448114014E+02
               85.0  -0.3563797363E+02
               90.0  -0.3739568848E+02
               95.0  -0.3635775146E+02
              100.1  -0.3782224121E+02
        >>>> The VARIANCE is :0.25  [hPa^2]
        <<<<
    <<<

 >> LIQUID FLOW rate measured at the bottom of the column
    >>> CONNECTION       : ELM50  BOT_1
        >>>> ANNOTTAION : OUTFLOW
        >>>> multiply measurements by FACTOR: -1.0E-06
        >>>> paired DATAset
             0.3000000000E+03    0.9869399946E+01  [mg/sec]
             0.6000000000E+03    0.1039689596E+02
             0.9000000000E+03    0.1162893932E+02
             0.1200000000E+04    0.1353439620E+02
             0.1500000000E+04    0.1469761628E+02
             0.1800000000E+04    0.1666155185E+02
             0.2100000000E+04    0.1897391849E+02
             0.2400000000E+04    0.1223393610E+02
             0.2700000000E+04    0.1029861869E+02
             0.3000000000E+04    0.9735449567E+01
             0.3600000000E+04    0.6692369425E+01
             0.3900000000E+04    0.8237884686E+01
             0.4200000000E+04    0.8513689863E+01
             0.4800000000E+04    0.6407594356E+01
             0.5100000000E+04    0.6761772056E+01
             0.5400000000E+04    0.6338728781E+01
             0.5700000000E+04    0.6168681921E+01
             0.6000000000E+04    0.4366627763E+01
        >>>> RELATIVE error is: 10.0 % of the individual measurement
        <<<<
    <<<
 <<     (terminate reading of observations)
```

```
> COMPUTATION (see Section 4.3.5)

  >> TOLERANCE

     >>> maximum number of ITERATIONs         : 3
     >>> maximum number of TOUGH2 calls       : 100 (-1)
     >>> ignore WARNINGS
     <<<


  >> JACOBIAN

     >>> increment FACTOR for derivatives     : 0.005
     >>> FORWARD differences, switch after    : 2 iterations to
         CENTERED finite difference quotient
     <<<


  >> Program OPTIONS

         solve DIRECT problem
     >>> LEVENBERG-MARQUART algorithm

/*
     >>> evaluate OBJECTIVE function at       : 15 locations
     this option was invoked to generate
     the database for Figure 19
*/
     <<<


  >> ERROR analysis

     >>> a POSTERIORI = estimated error variance
     >>> calculate HESSIAN matrix for error analysis
         check LINEARITY ASSUMPTION at: 95 % confidence level
         estimate SIGMAs after                : -5 iterations
     <<<


  >> OUTPUT

     >>> MINUTES
     >>> PLOT results after                   :  5 iterations
         Print JACOBIAN
         Print RESIDUALS
         Print OBJECTIVE function
     <<<
  <<
 <
```

Figure 17: Inverse problem: ITOUGH2 input deck

The corresponding ITOUGH2 output file is the following:

```
                              @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                              @                                              @
                              @  @@@  @@@@@   @@   @  @   @@@  @  @   @@@@   @
                              @   @     @    @  @  @  @  @     @  @  @    @  @
                              @   @     @    @  @  @  @  @ @@  @@@@     @@   @
                              @   @     @    @  @  @  @  @   @ @  @    @@    @
                              @  @@@    @     @@    @@    @@@  @  @  @@@@@@  @
                              @                                              @
                              @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

                  Version V2.2 (February 1, 1994) for IBM RS/6000, S. Finsterle


=======================================================================================================================
                                   >>>>>>>>>>>>> Wed Feb 16 10:33 <<<<<<<<<<<<<
=======================================================================================================================

PROBLEM TITLE: Synthetic experiment to demonstrate parameter estimation using ITOUGH2

=======================================================================================================================
TOUGH2 INPUT FILE >>>samdir<<<    ITOUGH2 INPUT FILE >>>saminv<<<    WORKING DIRECTORY >>>itough2<<<
=======================================================================================================================
| Equation of state package Nr. 3 is used.                                                                            |
| primary variables    #1     #2     #3     #4                                                                        |
| single-phase       : P      X      T                                                                                |
| two-phase          : Pg     Sg+10  T                                                                                |
=======================================================================================================================


*****  WARNING  *****
*  2 ambiguous keywords found on line  11:
*            >>>> ANNOTATION          : PORE SIZE INDEX
* Keyword No. 2 is interpreted!
*     1 : INDEX
* --> 2 : ANNOTATION
*****  WARNING  *****
```

```
--- Lines  31 to  37 skipped.

--- Lines 129 to 133 skipped.

========================================================================================================================
                                                             INPUT
========================================================================================================================

PARAMETERS
==========

------------------------------------------------------------------------------------------------------------------------
 # ID ANNOTATION       PARAMETER        V/L/F  ROCKS    INIT. GUESS     STD. DEV. LOWER BOUND  UPPER BOUND  PAR
------------------------------------------------------------------------------------------------------------------------
 1  4 PORE SIZE INDEX  CAP. PRESSURE    VALUE  MATRI     .23500E+01   .25000E+00   .10000E+01   .30000E+01   1
 2  4 AIR ENTRY PRES.  CAP. PRESSURE    VALUE  MATRI     .45000E-01   .10000E+00  -.10000E+51   .10000E+51   2
------------------------------------------------------------------------------------------------------------------------


OBSERVATIONS
============

Number of datasets                     :   2
Number of times                        :  20
Number of PRIOR INFO.                  :       2
Number of CAPILLARY PRES.              :      20
Number of FLOW RATE                    :      20
                                              ---
Total number of observations           :      42
                                              ===

TIMES [min]
-----------
  .5000E+01   .1000E+02   .1500E+02   .2000E+02   .2500E+02
  .3000E+02   .3500E+02   .4000E+02   .4500E+02   .5000E+02
  .5500E+02   .6000E+02   .6500E+02   .7000E+02   .7500E+02
  .8000E+02   .8500E+02   .9000E+02   .9500E+02   .1000E+03
```

```
-----------------------------------------------------------------------------------------------------------------------------
 N  ID  ANNOTATION        DATATYPE          ELEME/CONNE         STD. DEV.    MIN. TIME    MAX. TIME   V/M/S  DEFINED BY
-----------------------------------------------------------------------------------------------------------------------------
 1   5  TENSIOMETER       CAPILLARY PRES.   ELM 5              .50000E+02   .50000E+01   .10010E+03  VALUE  DATAPOINTS:  20
 2   6  OUTFLOW           LIQUID FLOW RAT   ELM50 BOT 1     REL. 10.00 %   .50000E+01   .10000E+03  VALUE  DATAPOINTS:  20
-----------------------------------------------------------------------------------------------------------------------------


COMPUTATIONAL PARAMETERS
========================

Total number of parameters to be estimated      :           2
Total number of observations                    :          42
Maximum number of iterations                    :           3
Maximum number of calls to TOUGH2               :         100
Increment factor for computing derivatives      :   .50000E-02
Maximum allowable step size                     :  -.99900E+03
Finite difference quotient for Jacobian         :  2 Forward -> Centered
Variance for error analysis                     : A posteriori
Plot format                                     : Tecplot
Library                                         : none (it2XXXX.f)
Computer version                                : IBM RS/6000
Optimization algorithm                          : Levenberg-Marquardt


--- End of ITOUGH2 input job:  147 lines read,  .650 CPU-seconds used
```

```
 =================================================================================================
                                                 OUTPUT
 =================================================================================================


 LEVENBERG-MARQUARDT ALGORITHM

>I = NEW ITERATION, J = JACOBIAN, S = STEP, U = UNSUCCESSFUL STEP, B = BOUNDS, M = MESSAGE, C = CONVERGENCE

 -------------------------------------------------------------------------------------------------
 ITER TOUGH2 OBJ FUNC. MAX. RESID. EQU. PORE SIZE INDEX AIR ENTRY PRES.
 -------------------------------------------------------------------------------------------------
>I  0    1  .13049E+04  .12259E+03   37     .235000E+01     .450000E-01
 J  1 Gradient     =    .37045E+06 Forward finite differences
 S    Step size    =    .84294E-01     Scaled step size =   .125146E+00  Levenberg parameter =  .10E-02
>I  1    5  .41430E+02  .85695E+01   42     .243412E+01     .503963E-01
 J  2 Gradient     =    .48253E+04 Forward finite differences
 S    Step size    =    .30990E-01     Scaled step size =   .133582E-01  Levenberg parameter =  .10E-03
>I  2    9  .37243E+02  .93329E+01   42     .246511E+01     .501924E-01
 J  3 Gradient     =    .45249E+04 Centered finite differences
 S    Step size    =    .15717E-02     Scaled step size =   .196481E-02  Levenberg parameter =  .10E-04
 U       1. unsuccessful step!              F(k+1)/F(k) =   .101456E+01  Levenberg parameter =  .10E-03
 S    Step size    =    .15706E-02     Scaled step size =   .196425E-02  Levenberg parameter =  .10E-03
 U       2. unsuccessful step!              F(k+1)/F(k) =   .101456E+01  Levenberg parameter =  .10E-02
 S    Step size    =    .15594E-02     Scaled step size =   .195871E-02  Levenberg parameter =  .10E-02
 U       3. unsuccessful step!              F(k+1)/F(k) =   .101455E+01  Levenberg parameter =  .10E-01
 S    Step size    =    .14520E-02     Scaled step size =   .190547E-02  Levenberg parameter =  .10E-01
 U       4. unsuccessful step!              F(k+1)/F(k) =   .101409E+01  Levenberg parameter =  .10E+00
 S    Step size    =    .69096E-03     Scaled step size =   .152642E-02  Levenberg parameter =  .10E+00
>I  3   19  .37039E+02  .93071E+01   42     .246442E+01     .502677E-01

 C   Maximum number of iterations reached. MITER =  3  --> Terminate!
```

```
Jacobian at the Solution (scaled)                                              Sensitivity coefficients
---------------------------------                                              ------------------------
       Time    #  Observation     PORE SIZE INDEX  AIR ENTRY PRES.            PORE SIZE INDEX  AIR ENTRY PRES.
 .30000E+03    3  TENSIOMETER          .46934E+01      -.65704E+03                 .23467E+03      -.32852E+05
 .30000E+03    4  OUTFLOW             -.15029E+00       .11533E+01                -.14832E-06       .11383E-05
 .60000E+03    5  TENSIOMETER         -.11923E+02      -.85312E+03                -.59615E+03      -.42656E+05
 .60000E+03    6  OUTFLOW             -.25804E+01       .73934E+00                -.26829E-05       .76869E-06
 .90000E+03    7  TENSIOMETER         -.15330E+02      -.95843E+03                -.76649E+03      -.47921E+05
 .90000E+03    8  OUTFLOW              .14880E+01       .73712E+01                 .17304E-05       .85719E-05
 .12000E+04    9  TENSIOMETER         -.17143E+02      -.10115E+04                -.85717E+03      -.50574E+05
 .12000E+04   10  OUTFLOW              .41674E+01      -.13184E+02                 .56403E-05      -.17843E-04
 .15000E+04   11  TENSIOMETER         -.17840E+02      -.10579E+04                -.89198E+03      -.52895E+05
 .15000E+04   12  OUTFLOW             -.38519E+00      -.76855E+00                -.56614E-06      -.11296E-05
 .18000E+04   13  TENSIOMETER         -.17937E+02      -.10981E+04                -.89683E+03      -.54907E+05
 .18000E+04   14  OUTFLOW             -.61505E+01       .19481E+01                -.10248E-04       .32458E-05
 .21000E+04   15  TENSIOMETER         -.17615E+02      -.11337E+04                -.88077E+03      -.56687E+05
 .21000E+04   16  OUTFLOW             -.54055E+02       .37320E+01                -.10256E-03       .70811E-05
 .24000E+04   17  TENSIOMETER         -.16660E+02      -.11675E+04                -.83298E+03      -.58376E+05
 .24000E+04   18  OUTFLOW             -.15344E+02       .29030E+00                -.18772E-04       .35515E-06
 .27000E+04   19  TENSIOMETER         -.16011E+02      -.12004E+04                -.80054E+03      -.60019E+05
 .27000E+04   20  OUTFLOW             -.54746E+01      -.62652E+00                -.56380E-05      -.64523E-06
 .30000E+04   21  TENSIOMETER         -.15941E+02      -.12320E+04                -.79703E+03      -.61599E+05
 .30000E+04   22  OUTFLOW             -.17237E+01      -.53830E+00                -.16781E-05      -.52406E-06
 .33000E+04   23  TENSIOMETER         -.15538E+02      -.12622E+04                -.77690E+03      -.63111E+05
 .33000E+04   24  OUTFLOW             -.19355E+01      -.17453E+00                -.16012E-05      -.14439E-06
 .36000E+04   25  TENSIOMETER         -.15285E+02      -.12912E+04                -.76423E+03      -.64558E+05
 .36000E+04   26  OUTFLOW             -.21813E+01       .31268E+00                -.14598E-05       .20926E-06
 .39000E+04   27  TENSIOMETER         -.15299E+02      -.13189E+04                -.76493E+03      -.65945E+05
 .39000E+04   28  OUTFLOW             -.15226E+01       .56495E+00                -.12543E-05       .46540E-06
 .42000E+04   29  TENSIOMETER         -.14908E+02      -.13455E+04                -.74539E+03      -.67275E+05
 .42000E+04   30  OUTFLOW             -.16350E+01       .73764E+00                -.13920E-05       .62800E-06
 .45000E+04   31  TENSIOMETER         -.14644E+02      -.13711E+04                -.73221E+03      -.68554E+05
 .45000E+04   32  OUTFLOW             -.19817E+01       .10142E+01                -.14082E-05       .72071E-06
 .48000E+04   33  TENSIOMETER         -.14601E+02      -.13957E+04                -.73007E+03      -.69786E+05
 .48000E+04   34  OUTFLOW             -.20713E+01       .11967E+01                -.13272E-05       .76683E-06
 .51000E+04   35  TENSIOMETER         -.14230E+02      -.14195E+04                -.71150E+03      -.70975E+05
 .51000E+04   36  OUTFLOW             -.20199E+01       .11595E+01                -.13658E-05       .78400E-06
 .54000E+04   37  TENSIOMETER         -.13967E+02      -.14425E+04                -.69837E+03      -.72124E+05
 .54000E+04  38  OUTFLOW            -.21382E+01      .12370E+01             -.13553E-05       .78412E-06
 .57000E+04  39  TENSIOMETER        -.13891E+02    -.14647E+04              -.69457E+03     -.73236E+05
 .57000E+04  40  OUTFLOW            -.20850E+01      .12559E+01             -.12862E-05       .77473E-06
 .60000E+04  41  TENSIOMETER        -.13544E+02    -.14863E+04              -.67718E+03     -.74314E+05
 .60000E+04  42  OUTFLOW            -.29520E+01      .17413E+01             -.12890E-05       .76038E-06
----------------------------------------------------------------
   Positive                          .10349E+02     .24454E+02
   Negative                          .40269E+03     .24193E+05
Prior info.                         -.40000E+01    -.10000E+02
----------------------------------------------------------------
   Total                             .40904E+03     .24207E+05
================================================================


Approximation of Hessian: H=(JT*P*J)
------------------------------------
                                   1               2

 1      PORE SIZE INDEX    .7878282E+04    .3565558E+06
 2      AIR ENTRY PRES.    .3565558E+06    .3012210E+08



========================================================================================================================
                                                ERROR ANALYSIS
========================================================================================================================


Error analysis is based on >>> a posteriori <<< variance:    .9259742E+00


Covariance(L+D)/Correlation(U) Matrix of Estimated Parameters
-------------------------------------------------------------
                    PORE SIZE INDEX  AIR ENTRY PRES.
PORE SIZE INDEX         .26648E-03      -.73193E+00
AIR ENTRY PRES.        -.31543E-05       .69696E-07
```

```
Standard Deviations
-------------------
PARAMETER          BEST ESTIAMTE  COND. P.D.F.  JOINT P.D.F.   COND./JOINT
PORE SIZE INDEX    .2464424E+01  .1112300E-01  .1632421E-01          .681
AIR ENTRY PRES.    .5026774E-01  .1798850E-03  .2640008E-03          .681


Eigenanalysis of Covariance Matrix
----------------------------------
Performance index      :   .37489587E-01
Condition number       :   .12139590E-03
Scaled condition number:   .29178028E+00


Eigenvalues
-----------
                                       1                2
                         PORE SIZE INDEX AIR ENTRY PRES.
 1      EIGENVALUE     :    .2665171E-03    .3235409E-07


Scaled Eigenvalues
------------------
                                       1                2
                         PORE SIZE INDEX AIR ENTRY PRES.
 1      EIGENVALUE     :    .4388281E-04    .1280414E-04


Eigenvectors
------------
                                       1                2
                         PORE SIZE INDEX AIR ENTRY PRES.
 1      PORE SIZE INDEX     .9999299E+00    .1183762E-01
 2      AIR ENTRY PRES.   -.1183762E-01    .9999299E+00

Trace(P*Q11) =    .200000000000E+01 = U =  2
```

```
=================================================================================================================
                                              RESIDUAL ANALYSIS
=================================================================================================================

RESIDUAL : observed - computed
R*P*R    : squared weighted residual
Yi       : local reliablity
Wi       : normalized residual
SDE      : smallest detectable error , alpha = .05, beta = .05, delta = 3.3114
STD. DEV.: a posteriori standard deviation of computed system response
*        : marks residuals for which abs(Wi) > u(0.95) = 1.6557

-----------------------------------------------------------------------------------------------------------------
 # OBSERVATION  AT TIME  MEASURED   COMPUTED    RESIDUAL       R*P*R         Yi          Wi         SDE  STD. DEV.*
-----------------------------------------------------------------------------------------------------------------
 1 PORE SIZE INDEX      .2350E+01  .2464E+01 -.1144E+00  .2095E+00  .9956E+00 -.4587E+00  .8297E+00  .1591E-01
 2 AIR ENTRY PRES.      .4500E-01  .5027E-01 -.5268E-02  .2775E-02  .1000E+01 -.5268E-01  .3311E+00  .2573E-03
 3 TENSIOMETER  .50E+01 -.1698E+04 -.1645E+04 -.5377E+02  .1156E+01  .9432E+00 -.1107E+01  .1705E+03  .1147E+02
 4 OUTFLOW      .50E+01 -.9869E-05 -.1022E-04  .3458E-06  .1227E+00  .1000E+01  .3503E+00  .3268E-05  .2582E-08
 5 TENSIOMETER  .10E+02 -.2076E+04 -.2140E+04  .6429E+02  .1653E+01  .9749E+00  .1302E+01  .1677E+03  .7618E+01
 6 OUTFLOW      .10E+02 -.1040E-04 -.1085E-04  .4484E-06  .1860E+00  .9982E+00  .4316E+00  .3446E-05  .4283E-07
 7 TENSIOMETER  .15E+02 -.2357E+04 -.2367E+04  .9610E+01  .3694E-01  .9652E+00  .1956E+00  .1685E+03  .8980E+01
 8 OUTFLOW      .15E+02 -.1163E-04 -.1158E-04 -.4541E-07  .1525E-02  .9995E+00 -.3906E-01  .3852E-05  .2596E-07
 9 TENSIOMETER  .20E+02 -.2529E+04 -.2524E+04 -.4979E+01  .9918E-02  .9587E+00 -.1017E+00  .1691E+03  .9775E+01
10 OUTFLOW      .20E+02 -.1353E-04 -.1253E-04 -.1005E-05  .5517E+00  .9949E+00 -.7446E+00  .4493E-05  .9315E-07
11 TENSIOMETER  .25E+02 -.2598E+04 -.2647E+04  .4918E+02  .9675E+00  .9551E+00  .1006E+01  .1694E+03  .1019E+02
12 OUTFLOW      .25E+02 -.1470E-04 -.1410E-04 -.5945E-06  .1636E+00  .1000E+01 -.4045E+00  .4867E-05  .8797E-08
13 TENSIOMETER  .30E+02 -.2689E+04 -.2750E+04  .6172E+02  .1524E+01  .9533E+00  .1264E+01  .1696E+03  .1040E+02
14 OUTFLOW      .30E+02 -.1666E-04 -.1681E-04  .1529E-06  .8420E-02  .9896E+00  .9224E-01  .5546E-05  .1637E-06
15 TENSIOMETER  .35E+02 -.2862E+04 -.2841E+04 -.2135E+02  .1823E+00  .9525E+00 -.4374E+00  .1696E+03  .1048E+02
16 OUTFLOW      .35E+02 -.1897E-04 -.1832E-04 -.6579E-06  .1202E+00  .1999E+00 -.7756E+00  .1405E-04  .1633E-05
17 TENSIOMETER  .40E+02 -.2988E+04 -.2927E+04 -.6110E+02  .1493E+01  .9525E+00 -.1252E+01  .1696E+03  .1048E+02
18 OUTFLOW      .40E+02 -.1223E-04 -.1162E-04 -.6165E-06  .2540E+00  .9356E+00 -.5210E+00  .4188E-05  .2987E-06
19 TENSIOMETER  .45E+02 -.3017E+04 -.3010E+04 -.6621E+01  .1753E-01  .9513E+00 -.1358E+00  .1698E+03  .1062E+02
20 OUTFLOW      .45E+02 -.1030E-04 -.9819E-05 -.4797E-06  .2169E+00  .9918E+00 -.4677E+00  .3424E-05  .8958E-07
21 TENSIOMETER  .50E+02 -.3111E+04 -.3091E+04 -.2053E+02  .1685E+00  .9491E+00 -.4214E+00  .1700E+03  .1085E+02
22 OUTFLOW      .50E+02 -.9735E-05 -.8847E-05 -.8882E-06  .8324E+00  .9992E+00 -.9127E+00  .3225E-05  .2660E-07
23 TENSIOMETER  .55E+02 -.3178E+04 -.3168E+04 -.1079E+02  .4659E-01  .9470E+00 -.2218E+00  .1701E+03  .1108E+02
```

```
24  OUTFLOW      .55E+02 -.8274E-05 -.8209E-05  -.6348E-07   .5887E-02   .9990E+00  -.7677E-01   .2741E-05   .2545E-07
25  TENSIOMETER  .60E+02 -.3274E+04 -.3241E+04  -.3310E+02   .4383E+00   .9447E+00  -.6812E+00   .1704E+03   .1132E+02
26  OUTFLOW      .60E+02 -.6692E-05 -.7736E-05   .1043E-05   .2430E+01   .9987E+00   .1560E+01   .2218E-05   .2327E-07
27  TENSIOMETER  .65E+02 -.3261E+04 -.3312E+04   .5060E+02   .1024E+01   .9422E+00   .1043E+01   .1706E+03   .1156E+02
28  OUTFLOW      .65E+02 -.8238E-05 -.7354E-05  -.8835E-06   .1150E+01   .9994E+00  -.1073E+01   .2729E-05   .2005E-07
29  TENSIOMETER  .70E+02 -.3349E+04 -.3379E+04   .3084E+02   .3804E+00   .9396E+00   .6363E+00   .1708E+03   .1182E+02
30  OUTFLOW      .70E+02 -.8514E-05 -.7030E-05  -.1483E-05   .3028E+01   .9993E+00  -.1743E+01   .2820E-05   .2227E-07*
31  TENSIOMETER  .75E+02 -.3449E+04 -.3444E+04  -.4158E+01   .6916E-02   .9369E+00  -.8591E-01   .1711E+03   .1209E+02
32  OUTFLOW      .75E+02 -.7106E-05 -.6751E-05  -.3549E-06   .2495E+00   .9989E+00  -.4998E+00   .2354E-05   .2254E-07
33  TENSIOMETER  .80E+02 -.3448E+04 -.3507E+04   .5900E+02   .1392E+01   .9343E+00   .1221E+01   .1713E+03   .1233E+02
34  OUTFLOW      .80E+02 -.6408E-05 -.6500E-05   .9287E-07   .2101E-01   .9988E+00   .1450E+00   .2123E-05   .2126E-07
35  TENSIOMETER  .85E+02 -.3564E+04 -.3568E+04   .3757E+01   .5645E-02   .9313E+00   .7786E-01   .1716E+03   .1261E+02
36  OUTFLOW      .85E+02 -.6762E-05 -.6274E-05  -.4879E-06   .5207E+00   .9989E+00  -.7220E+00   .2240E-05   .2188E-07
37  TENSIOMETER  .90E+02 -.3740E+04 -.3620E+04  -.1193E+03   .5163E+01   .9283E+00  -.2477E+01   .1718E+03   .1288E+02*
38  OUTFLOW      .90E+02 -.6339E-05 -.6067E-05  -.2721E-06   .1842E+00   .9987E+00  -.4295E+00   .2100E-05   .2171E-07
39  TENSIOMETER  .95E+02 -.3636E+04 -.3682E+04   .4671E+02   .8728E+00   .9255E+00   .9711E+00   .1721E+03   .1313E+02
40  OUTFLOW      .95E+02 -.6169E-05 -.5876E-05  -.2929E-06   .2255E+00   .9988E+00  -.4751E+00   .2044E-05   .2061E-07
41  TENSIOMETER  .10E+03 -.3779E+04 -.3737E+04  -.4208E+02   .7084E+00   .9222E+00  -.8765E+00   .1724E+03   .1342E+02
42  OUTFLOW      .10E+03 -.4367E-05 -.5955E-05   .1589E-05   .9307E+01   .9976E+00   .3643E+01   .1448E-05   .2065E-07*

Iteration Statistics
--------------------
Number of iterations              :    3
Number of TOUGH2 calls            :   20
Number of Jacobians evaluated     :    3
Maximum residual at function      :   42


Error Variances
---------------
Estimated error variance          :   .9747097E+00
Variance used in error analysis   :   .9747097E+00
```

```
Objective Function
------------------
Initial value of objective function :    .1304934E+04
Minimum value of objective function :    .3703897E+02            100.00 %
Log-likelihood                      :    .1142298E+03
Likelihood                          :    .1567881E-24


-------------------------------------------------------------------------
Mean Absolute Residual (MAR) and Contribution to Objective Function (COF)
-------------------------------------------------------------------------
For Each Dataset        Datapoints                MAR  UNITS            COF
-------------------------------------------------------------------------
TENSIOMETER       :             20          .3739E+02  [Pa]         46.57 %
OUTFLOW           :             20          .5770E-06  [kg/sec]     52.86 %
-------------------------------------------------------------------------
For Each Datatype       Datapoints                MAR  UNITS            COF
-------------------------------------------------------------------------
PRIOR INFORMATION :              2                                    .57 %
CAPILLARY PRES.   :             20          .3739E+02  [Pa]         46.57 %
FLOW RATE         :             20          .5770E-06  [kg/sec]     52.86 %
-------------------------------------------------------------------------


Fisher Model Test
-----------------
Estimated error variance             :    .9747097E+00
Critical value of F-distribution     :    .1402846E+01
Degree of freedom                    :              38    (No prior information)
Confidence level (1-alpha)           :            95.0 [%]
Confidence level (1-beta)            :            95.0 [%]
Warning                              : Accuracy underestimated!
Error analysis based on              : a posteriori variance =  .9747097E+00
```

```
Model Structure Criteria
------------------------
AIC=-2ln(S)+2*n                   :    .1182298E+03 (Akaike)
BIC=-2ln(S)+n*ln(m)               :    .1217051E+03 (Akaike, Rissanen, Schwarz)
PHI=-2ln(S)+2*n*ln[ln(m)]         :    .1195037E+03 (Hannan)
DM =-2ln(S)+n*ln(m/2PI)+ln|F|     :    .1434548E+03 (Kashyap)




?!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
ESTIMATED PARAMETER  V/L/F   ROCKS   PAR  INIT. GUESS  BEST ESTIMATE          STANDARD DEVIATIONS
                                                                   A PRIORI  CONDITIONAL JOINT P.D.F. SENSITIVITY
PORE SIZE INDEX      VALUE   MATRI     1  .235000E+01   .2464424E+01 .2500000E+00 .1112300E-01 .1632421E-01       409.0
AIR ENTRY PRES.      VALUE   MATRI     2  .450000E-01   .5026774E-01 .1000000E+00 .1798850E-03 .2640008E-03     24207.1
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


--- End of ITOUGH2 simulation job --- elapsed time =  493.5     sec.

    Error  >> Subroutine <<  Message
     2594  >>> FINDKEY  <<<  WARNING              --> Ambiguous keywords

---     0 error(s) and  1 warning(s) detected
```

```
==========================================================================================
    PROGRAM  VERSION   DATE                COMMENT
------------------------------------------------------------------------------------------
    ITOUGH2  2.2                           Current version: V2.2 (FEBRUARY  1, 1994)
------------------------------------------------------------------------------------------
    ITOUGH2  1.0      1 SEPTEMBER 1992     First version is an adaptation of ITOUGH V1.0
    ITOUGH2  1.1      1 JANUARY   1993     INOBSDAT,INPAIRED: Flexible time specification
                      1 JANUARY   1993     INERROR,MTCARLO: number of classes can be specified
                      1 FEBRUARY  1993     INUSER,USERPAR: user specified parameter
                      1 FEBRUARY  1993     INUSROBS,USEROBS: user specified observations
                     15 FEBRUARY  1993     PLOTFI,PLOTCHAR: Reformat plotfiles, plot characteristic curves
                     15 FEBRUARY  1993     PLOTCHAR: Plots characteristic curves
    ITOUGH2  1.2      1 APRIL     1993     Add version for IBM RS/6000
                     26 MAY       1993     INANNEAL,ANNEAL: Simulated Annealing minimization
    ITOUGH2  2.0     12 AUGUST    1993     File t2cg1.f: Conjugate gradient solvers added
    ITOUGH2  2.1     23 SEPTEMBER 1993     Rearrange parameter vector
                     29 SEPTEMBER 1993     Rearrange observation vector
                     15 FEBRUARY  1994     Add new observation types
    ITOUGH2  2.2      1 FEBRUARY  1994     Steady-state data points allowed
                                           This version is documented in LBL-34581, ITOUGH2 User's Guide
------------------------------------------------------------------------------------------
    WHATCOM  1.0     10 AUGUST    1993     #35: Q: WHAT COMPUTER IS USED? A: IBM
    CPUSEC   1.0     10 AUGUST    1993     #--: RETURNS CPU-TIME (VERSION IBM)
    OPENFILE 1.2      5 AUGUST    1993     #31: OPENS MOST OF THE FILES
    LENOS    1.0      1 AUGUST    1992     #28: RETURNS LENGTH OF LINE
    PREC     1.0      1 AUGUST    1992     #86: CALCULATE MACHINE DEPENDENT CONSTANTS
    ITHEADER 1.0      1 AUGUST    1992     #29: PRINTS ITOUGH2 HEADER
    DAYTIM   1.0     10 AUGUST    1993     #32: RETURNS DATE AND TIME (VERSION IBM)
    THEADER  1.1     27 MAY       1993     #30: PRINTS TOUGH2 HEADER
    INPUT    1.1     15 MAY       1993     READ ALL DATA PROVIDED THROUGH FILE *INPUT*, NEW:ELEM2/CONN2
    FLOPP    1.0     11 APRIL     1991     CALCULATE NUMBER OF SIGNIFICANT DIGITS FOR FLOATING POINT ARITHMETIC
    RFILE    1.0     23 APRIL     1991     INITIALIZE DATA FROM FILES *MESH* OR *MINC*, *GENER*, AND *INCON*
    INDATA   1.0      5 MARCH     1991     PROVIDE PRINTOUT OF MOST DATA PROVIDED THROUGH FILE *INPUT*
    ITINPUT  1.0      1 AUGUST    1992     # 2: READS COMMANDS OF COMMAND LEVEL 1
    READCOMM 1.0      1 AUGUST    1992     #24: READS A COMMAND
    FINDKEY  1.1      4 AUGUST    1993     #25: READS A KEYWORD
    LTU      1.0      1 AUGUST    1992     #26: CONVERTS LOWER TO UPPER CASE
    INPARAME 2.1     21 SEPTEMBER 1993     # 3: READS PARAMETERS TO BE ESTIMATED
```

```
INPAR    2.1    21 SEPTEMBER 1993    # 4: READS PARAMETER VALUES, WEIGHTS, ETC.
INELEM   2.2    11 MARCH     1994    #23: READS GRID BLOCK NAME AFTER A COLON
NEXTWORD 1.0     1 AUGUST    1992    #27: EXTRACTS NEXT WORD ON A LINE
INWBP    2.1    21 SEPTEMBER 1993    #11: READS WEIGHT, BOUNDS, ANNOTATION, AND PARAMETERS
ERROR    1.0     1 AUGUST    1992    #34: PRINTS ERROR MESSAGES
INOBSERV 2.2    14 FEBRUARY  1994    #12: READS TYPE OF OBSERVATION
INTIMES  1.0     1 AUGUST    1992    #13: READS TIMES AT WHICH OBSERVATIONS ARE AVAILABLE
INOBS    2.2    14 FEBRUARY  1994    #15: READS OBSERVATION INFOS
INOBSDAT 2.2    27 JANUARY   1994    #17: READS ALL OBSERVED DATA
INPAIRED 2.2    14 JANUARY   1994    #19: READS PAIRED DATA SET
INWEIGHT 1.1     5 AUGUST    1993    #20: READS WEIGHTS
INCOMPUT 1.0     1 AUGUST    1992    #16: READS VARIOUS COMPUTATIONAL PARAMETERS
INTOLER  2.2    15 FEBRUARY  1994    #83: READS TOLERANCE/STOPPING CRITERIA
READINT  1.0     1 AUGUST    1992    #21: RAEDS AN INTEGER AFTER A COLON
INJACOB  1.0     1 AUGUST    1992    #84: READS PARAMETERS FOR COMPUTING JACOBIAN
READREAL 1.0     1 AUGUST    1992    #22: READS A REAL AFTER A COLON
INOPTION 1.0     1 AUGUST    1992    #85: READS PROGRAM OPTIONS
INERROR  1.2    10 AUGUST    1993    #81: READS COMMANDS FOR ERROR ANALYSIS
INPRINT  2.2    10 MARCH     1994    #80: READS OUTPUT OPTIONS
GETINDEX 2.2    11 MARCH     1994    #45: GETS INDEX OF ELEMENTS, CONNECTIONS, AND SOURCES
INIGUESS 2.1    21 SEPTEMBER 1993    #38: INITIAL GUESS OF PARAMETERS (XGUESS)
GETNMAT  2.1    21 SEPTEMBER 1993    #44: IDENTIFIES MATERIAL NUMBER
IXLBXUB  2.1    21 SEPTEMBER 1993    #43: INITIALIZES ARRAY XLB AND XUB
SETWSCAL 2.1    29 SEPTEMBER 1993    #39: INITIALIZES ARRAY WSCALE
OBSMEAN  1.0     1 AUGUST    1992    #40: CALCULATES MEAN OF OBSERVATIONS
OBSERVED 2.1    15 NOVEMBER  1993    #78: RETURNS OBSERVED DATA AS A FUNCTION OF TIME
SETPLL   1.1    10 MARCH     1993    #41: INITIALIZES MATRIX PLL AND ARRAY WSCALE
SETXSCAL 1.0     1 AUGUST    1992    #42: INITIALIZES ARRAY XSCALE
IN_OUT   2.2    14 FEBRUARY  1994    #35: PRINTS A SUMMARY OF INPUT DATA
TIMEWIND 1.0     1 AUGUST    1992    #53: SETS TIME WINDOW
LEVMAR   2.1    16 SEPTEMBER 1993    #99: LEVENBERG-MARQUARDT OPTIMIZATION ALGORITHM
FCNLEV   1.0     1 AUGUST    1992    #50: RETURNS WEIGHTED RESIDUAL VECTOR
UPDATE   2.2    13 DECEMBER  1993    #37: UPDATES PARAMETERS
PRIORINF 2.1    21 SEPTEMBER 1993    #48: PRIOR INFORMATION
OBSERVAT 2.2    15 FEBRUARY  1994    #62: COMPARES MEASURED AND CALCULATED QUANTITIES
GETMESH  1.1    15 APRIL     1993    #47: READS FILE MESH, MINC, GENER, AND INCON
GETINCON 2.1    26 SEPTEMBER 1993    #46: READS FILE INCON
INITTOUG 1.0     1 AUGUST    1992    #54: INITIALIZES TOUGH2 RUN (REPLACES CYCIT)
EOS      1.0    28 MARCH     1991    *EOS3* ... THERMOPHYSICAL PROPERTIES MODULE FOR WATER/AIR
```

```
    SAT      1.0       22 JANUARY   1990      STEAM TABLE EQUATION: SATURATION PRESSURE AS FUNCTION OF TEMPERATURE
    PCAP     1.2       10 NOVEMBER  1993      CAPILLARY PRESSURE S. FINSTERLE
    PP       1.0        1 FEBRUARY  1990      CALCULATE VAPOR PRESSURE, DENSITY, INT. ENERGY AS F(P,T,X)
    VISCO    1.0        1 FEBRUARY  1990      CALCULATE VISCOSITY OF VAPOR-AIR MIXTURES
    COVIS    1.0        1 FEBRUARY  1990      COEFFICIENT FOR GAS PHASE VISCOSITY CALCULATION
    VISS     1.0       22 JANUARY   1990      VISCOSITY OF VAPOR AS FUNCTION OF TEMPERATURE AND PRESSURE
    VISW     1.0       22 JANUARY   1990      VISCOSITY OF LIQUID WATER AS FUNCTION OF TEMPERATURE AND PRESSURE
    COWAT    1.0       22 JANUARY   1990      LIQUID WATER DENSITY AND INT. ENERGY AS FUNCTION OF TEMP. AND PRESSURE
    BALLA    1.0        5 MARCH     1991      PERFORM SUMMARY BALANCES FOR VOLUME, MASS, AND ENERGY
    CALLTOUG 1.0        1 AUGUST    1992      #55: CALLS TOUGH2 FOR ONE TIME STEP
    TSTEP    1.0        4 MARCH     1991      ADJUST TIME STEPS TO COINCIDE WITH USER-DEFINED TARGET TIMES
    MULTI    1.0        9 MAY       1991      ASSEMBLE ALL ACCUMULATION AND FLOW TERMS
    LINEQ    0.9 CG    23 April     1993      Interface for linear equation solvers (MA28 or conjugate gradient)
    SUPST    1.0       29 JANUARY   1990      VAPOR DENSITY AND INTERNAL ENERGY AS FUNCTION OF TEMPERATURE AND PRESSURE
    RELP     1.1       11 MARCH     1992      RELATIVE PERMEABILITIES S. FINSTERLE
    CONVER   1.0        4 MARCH     1991      UPDATE PRIMARY VARIABLES AFTER CONVERGENCE IS ACHIEVED
    OUT      1.0        5 MARCH     1991      PRINT RESULTS FOR ELEMENTS, CONNECTIONS, AND SINKS/SOURCES
    OBJFUN   2.1       29 SEPTEMBER 1993      #49: COMPUTE OBJECTIVE FUNCTION
    PLOTFILE 2.1       29 SEPTEMBER 1993      #58: WRITES PLOTFILE IN PLOPO-FORMAT
    RESIDUAL 2.1       29 SEPTEMBER 1993      #90: BACKCALCULATES RESIDUALS
    JAC      2.1       21 SEPTEMBER 1992      #51: CALCULATES FINITE DIFFERENCE JACOBIAN
    TERMINAT 2.2        3 JANUARY   1994      #61: PERFORM ERROR ANALYSIS AND TERMINATE ITOUGH2
    QFISHER  2.2       16 FEBRUARY  1994      #77: RETURNS QUANTILE OF F-DISTRIBUTION
    QCHI     1.0        1 AUGUST    1992      #88: RETURNS CHI-SQUARE QUANTILE
    POLYNOM  1.0        1 AUGUST    1992      #89: EVALUATES POLYNOM
    EIGEN    2.1       21 SEPTEMBER 1993      #59: PERFORMS EIGENANALYSIS
    BESTRUN  2.1       21 SEPTEMBER 1993      #56: FINAL TOUGH2 RUN
    WRIFI    1.0       22 JANUARY   1990      AT THE COMPLETION OF A TOUGH2 RUN, WRITE PRIMARY VARIABLES ON FILE *SAVE*
    LOGLIKE  2.1       29 SEPTEMBER 1993      #68: COMPUTE LOG-LIKELIHOOD
    MLLAMBDA 2.2       14 FEBRUARY  1994      #67: ESTIMATES NEW LAMBDAS
    QNORMAL  1.0        1 AUGUST    1992      #87: RETURNS QUANTILE OF NORMAL DISTRIBUTION
    PLOTIF   1.0       15 FEBRUARY  1993      #96: PLOT INTERFACE
    REFORMAT 1.1       15 APRIL     1993      #97: REFORMATS PLOT FILES
    QUOTES   1.0       15 FEBRUARY  1993      #98: RETURNS TEXT BETWEEN QUOTES
==========================================================================================================================
```

Figure 18: Inverse problem: ITOUGH output file

Figure 19 depicts the solution path of the Levenberg-Marquardt algorithm in the two-dimensional parameter space. The contours show the convexity of the objective function. The estimated error variance is close to one which implies that the *a priori* defined error structure is consistent with the *a posteriori* calculated residuals. The 95%-confidence region around the parameter set after 3 iterations (n=2.46, -1/α=5.03) is based on a first order error analysis. The orientation of the axis is given by the eigenvectors of the covariance matrix C_p, the length is proportional to the square-root of the eigenvalues, the factor of proportionality for the 95-% confidence level is $\sqrt{2 \cdot F_{2,40,0.95}} = 2.45$; the true parameter set (n=2.50, -1/α=5.0 bar) is within the estimated confidence region. The fact that the minimum of the objective function does not coincide with the true parameter combination reveals the bias invoked by the noise in the data. A rather detailed discussion of a similar solution is given in *Finsterle* [1993].

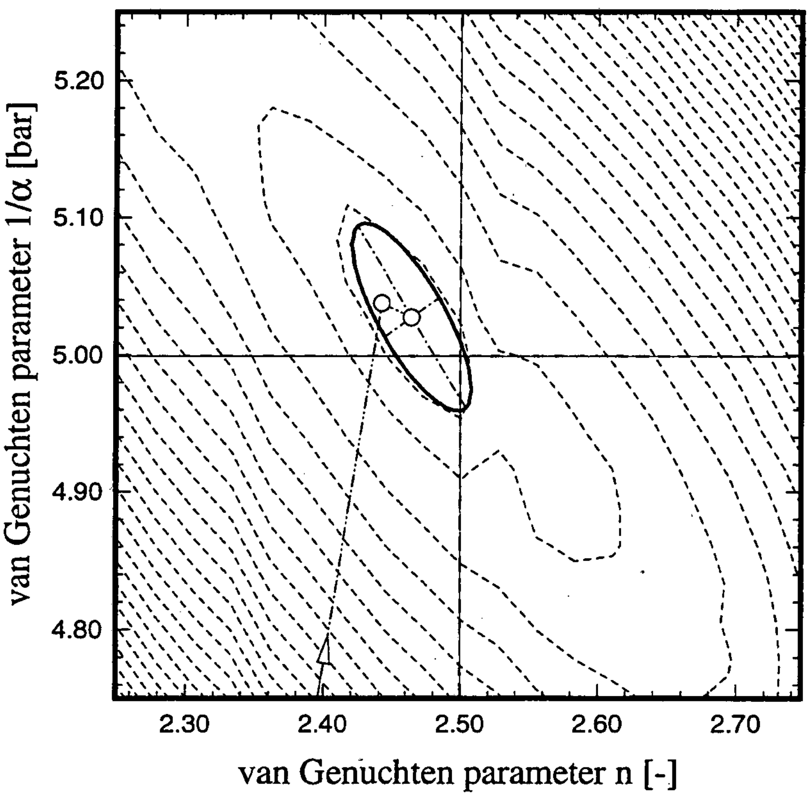


Figure 19: Objective function, solution path, and 95% confidence region

5.2 Estimation of Prediction Error

The impact of parameter uncertainties on the result of a prediction model may be studied by means of First-Order-Second-Moment or Monte-Carlo error analysis. In order to compare both methods, the errors of the four parameters are assumed to be uncorrelated. The appropriate ITOUGH2 input file is shown below:

```
ITOUGH2 input file to estimate prediction error by means of
1. FOSM (First-Order-Second-Moment error analysis)
2. Monte-Carlo simulations


  > PARAMETER
    >> ABSOLUTE permeability
       >>> ROCK type              : MATRI
           >>>> PARAMETER         : 1
           >>>> VARIANCE          : 0.38124E-04
           >>>> BOUNDS            : -16.5 -13.5
           >>>> GUESS             : -15.0
           the following two lines cause ITOUGH2 to generate
           log-normally distributed permeabilities (default)
           (only for Monte-Carlo)
           >>>> LOGARITHM
           >>>> NORMALly distributed
           <<<<
       <<<

    >> RELATIVE permeability (van Genuchten's functions)
       >>> ROCK type              : MATRI
           >>>> PARAMETER         : 1 (Residual liquid saturation)
           >>>> VARIANCE          : 0.29149E-03
           >>>> BOUNDS            : 0.01 0.75
           >>>> GUESS             : 0.30
           <<<<
       <<<
      >> CAPILLARY pressure (van Genuchten's function)
    >>> ROCK TYPE              : MATRI
        >>>> PARAMETER         : 1 (pore size distribution index)
        >>>> VARIANCE          : 0.12657E-02
        >>>> BOUNDS            : 2.0  5.0
        >>>> GUESS             : 2.5
        >>>> NORMALly distributed (default, only for Monte-Carlo)
        <<<<
    <<<

 >> CAPILLARY pressure (van Genuchten's function)
    >>> ROCK TYPE              : MATRI
        >>>> PARAMETER         : 2 (Air entry pressure [bar])
        >>>> VALUE
        >>>> VARIANCE          : 0.51199
        >>>> BOUNDS            : 0.0  1.0
        >>>> GUESS             : 0.05
        <<<<
    <<<
 <<

> OBSERVATIONS
 (points in space and time at which uncertainty of predicted
  system state is to be calculated)
 >> TIMES: 20  EQUALLY spaced
    6300.0 12000.0 sec
 >> LIQUID FLOW rate
    >>> CONNECTION : ELM50  BOT 1
        >>>> this is a dummy DATA set
             0.630000000000E+04  1.000000000000E+00
             0.120000000000E+05  1.000000000000E+00
        >>>> AUTOmatic error assignment
        <<<<
    <<<
 <<
```

```
  > COMPUTATION

    >> STOPPING criteria
       >>> maximum number of TOUGH2 calls: 400
       <<<

    >> ERROR analysis
       >>> FOSM error analysis
           if no covariance matrix is provided here, ITOUGH2
           takes the variances as specified in block PARAMETERS
/*
       >>> MONTE-CARLO simulations; the SEED number is: 11
           invoke this option to perform 400 Monte-Carlo simulations
*/
       <<<

    >> JACOBIAN (only needed for FOSM analysis)
       >>> increment FACTOR                : 0.005
       >>> CENTERED finite difference quotient
       <<<
    <<
  <
```

Figure 20: FOSM and Monte-Carlo: ITOUGH2 input file

The corresponding TOUGH2 input file allows simulation of the injection of liquid water into the partially saturated rock sample which is the final condition of the experiment described in Section 5.1. The system state of interest for which the uncertainty is to be calculated is the outflow of water at the bottom of the column. The result of both the FOSM and Monte-Carlo error analysis is shown in Figure 21:

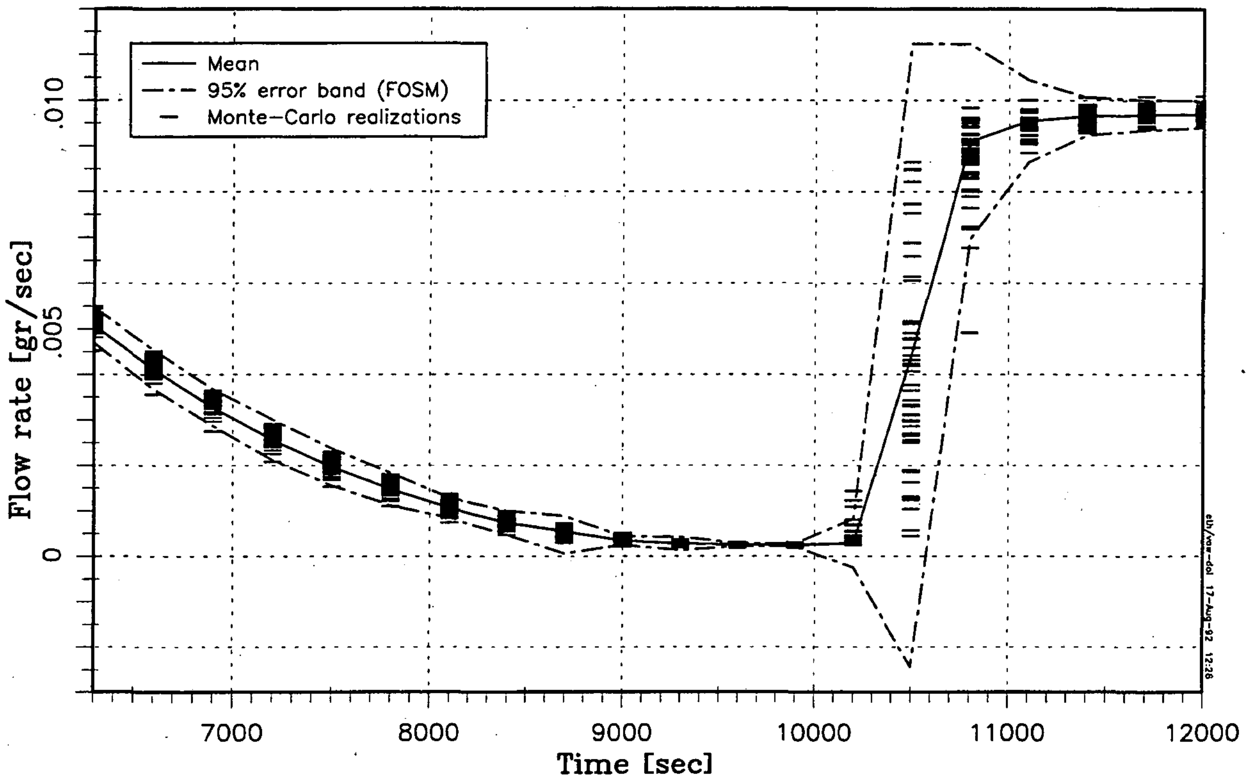


Figure 21: Predicted system response, error band, and Monte-Carlo realizations

It can be seen that the simple linear error analysis provides a rather good estimate of the 95%-confidence region at the beginning and the end of the test period. This is not true in the interval between 10,000 and 11,000 seconds where the breakthrough of the liquid front at the bottom of the column is a nonlinear function of the parameters. While the FOSM error analysis predicts a symmetric error band around the mean (leading to unphysical negative flow rates), the Monte-Carlo simulations provide a better estimate of the corresponding probability density function. More details can be found in *Finsterle* [1993].

Acknowledgment

ITOUGH2 (up to version 1.1) was developed at the Laboratory of Hydraulics, Hydrology, and Glaciology (VAW) at the Swiss Federal Institute of Technology (ETH), Zürich, in collaboration with the Swiss National Cooperative for the Disposal of Radioactive Waste, Nagra, Wettingen. This work was carried out under U.S. Department of Energy Contract No. DE-AC03-76SF00098 for the Director, Office of Civilian Radioactive Waste Management, Office of External Relations, and was administered by the Nevada Operations Office, U.S. Department of Energy, in cooperation with the Swiss National Cooperative for the Disposal of Radioactive Waste (Nagra), and was supported, in part, by the WIPP project, Sandia National Laboratories, under Document No. 129847. I would like to thank Karsten Pruess, Lawrence Berkeley Laboratory, for introducing me to two-phase flow modeling. I would also like to thank C. Doughty, U. Kuhlmann, S. Mishra, C. Zurbrügg, and Y. Tsang for their careful review of this manual.

References

Akaike, H., A New Look at Statistical Model Identification, IEEE Trans. Automat. Contr., AC-19, 716-722, 1974.

Baarda,W., A Testing Procedure for Use in Geodetic Networks, Netherlands Geodetic Commission, Publications on Geodesy, 2/5, 1968.

Carosio, A., Robuste Ausgleichung, Vermessung, Photogrammetrie, Kulturtechnik, 11, 293-297, 1979.

Carrera, J., Estimation of Aquifer Parameters Under Transient and Steady State Conditions, Ph.D. dissertation, Dep. of Hydrology and Water Resources, University of Arizona, Tucson, 1984.

Carrera, J., S.P. Neuman, Estimation of Aquifer Parameters Under Transient and Steady-State Conditions, Water Resources Research, 22 (2), 199-210, 1986.

Finsterle, S., Inverse Modellierung zur Bestimmung hydrogeologischer Parameter eines Zweiphasensystems, Mitteilung Nr. 121 der Versuchsanstalt für Wasserbau, Hydrologie und Glaziologie der Eidgenössischen Technischen Hochschule, Zürich, Switzerland, 1993.

Luckner, L., M. Th. van Genuchten, D. Nielsen, A Consistent Set of Parametric Models for the Two-Phase Flow of Immiscible Fluids in the Subsurface, Water Resources Research, 25 (10), 2187-2193, 1989.

Marquardt, D.W., An Algorithm for Least Squares Estimation of Nonlinear Parameters, SIAM J. Appl. Math., 11, 431-441, 1963.

Press, W.H., S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in FORTRAN, Second Edition, Cambridge University Press, 1992.

Pruess, K., TOUGH User's Guide, Nuclear Regulatory Commission, report NUREG/CR-4645 (also Lawrence Berkeley Laboratory, report LBL-20700, Berkeley, CA), 1987.

Pruess, K., TOUGH2 - A General-Purpose Numerical Simulator for Multiphase Fluid and Heat Flow, Lawrence Berkeley Laboratory, report LBL-29400, Berkeley, CA, 1991.

Scales, L.E., Introduction to Non-Linear Optimization, Springer-Verlag, New York, 1985.

Appendix A1: Adding a new parameter type

The purpose of this Appendix is to enable ITOUGH2 users to add a new parameter type to the permanent list of TOUGH2 input parameters to be estimated. Note that the user specified parameters defined by means of subroutine `USERPAR` (see Section 4.3.3, command `>> USER`) is designed for a quick enhancement of the code's ability to estimate specific parameters for a specific application. It is therefore of a temporary nature. If the user wishes to add a parameter type he or she expects to estimate in many applications, a permanent change of the code is advisable. This prevents an overloading of subroutine USERPAR and allows to check the input for consistency.

The following changes have to be made:

(1) *Subroutine INPARAME*:
This subroutine reads the commands for parameter definition. Increase parameter `NC` by one and add appropriate second level keyword in slot `NC-4` of array `COMMANDS`. Add a block similar to the following example:

```
C
C --- Estimate (new parameter)
         IVLF=120
         IDP=ICOMMAND
         CALL INPAR(IDP,IP,IVLF,LINE)
```

`IDP` is the identification number of the new parameter type. The variable `IVLF` indicates whether the value, factor, or logarithm of the corresponding parameter can be estimated and which shall be the default option:

`I00`	=	value
`0I0`	=	logarithm
`00I`	=	factor
`II0`	=	value or logarithm
`I0I`	=	value or factor
`0II`	=	logarithm or factor
`III`	=	value or logarithm or factor

The largest value for I indicates the default option. In our example (IVLF=120), either the value of the parameter or its logarithm can be estimated. Estimating a multiplication factor is not allowed and will be rejected if requested by the user. Estimating the logarithm is the default option.

More sophisticated input is possible including additional keywords and parameters to be transferred (see e.g. specification of initial conditions).

(2) *Subroutine IN_OUT*:
Add default annotation CPAR of new observation type in loop 1000.

(3) *Subroutine INIGUESS*:
In this subroutine, the parameter value from the TOUGH2 input deck is transferred to ITOUGH2 as an initial guess and starting point for the optimization. Assign the appropriate TOUGH2 parameter to the ITOUGH2 variable XGUESS. Provide a default parameter annotation (variable APAR). For details see examples in subroutine INIGUESS.

(4) *Subroutine UPDATE*:
In this subroutine, the TOUGH2 parameter is updated (see Figure 8). Assign the variable X to the corresponding TOUGH2 parameter in loop 1000 (see examples in subroutine UPDATE).

It is strongly suggested to update the version control statements at the beginning of each updated subroutine. Furthermore, the syntax of the new command should be documented and sent to the author of this manual together with a listing of the updated subroutines INIGUESS and UPDATE to make it available for other users.

Appendix A2: Adding a new observation type

The purpose of this Appendix is to enable ITOUGH2 users to add a new observation type to the permanent list of potential data being used for calibration. Note that the user specified observations defined by means of subroutine `USEROBS` (see Section 4.3.4, command `>> USER`) is designed for a quick enhancement of the code's ability to handle specific data for a specific application. It is therefore of a temporary nature. If the user wishes to add an observation type he or she expects to use more frequently, a permanent change of the code is advisable. This not only prevents an overloading of subroutine `USEROBS`, but also allows to check the input for consistency, and, most important, enables updating the error variance for this new observation type.

The following changes have to be made:

(1) Enhance parameter `MTYPE` by one, e.g. replace `MTYPE=9` by `MTYPE=10`.

(2) *Subroutine* *`INOBSERV`*:
This subroutine reads the commands for data definition. Increase parameter `NC` by one and add an appropriate second level keyword in slot `NC-1` of array `COMMANDS`. Add a block similar to the following example:

```
C
C --- (New data type)
      ELSE IF (ICOMMAND.EQ.9) THEN
        IDO=10
        IEC=2
        CALL INOBS(LINE,IDO,IO,IEC)
```

Here, `IDO` is an identification number of the new observation type (equal to the current value of `MTYPE`). Set `IEC` to 1 if the observation refers to a grid block, and 2 if it refers to a connection. More sophisticated input is possible including additional keywords and parameters to be transferred (see e.g. flow rates).

(3) *Subroutines* *`IN_OUT, MLLAMBDA, EVALOBJF, TERMINAT`*:
Add string describing new observation type in slot `MTYPE` of array `CTYP`.

(4) *Subroutine TERMINAT*:
Add string describing units of new observation type in slot `MTYPE` of array `CUNIT`.

(5) *Subroutine IN_OUT*:
Add default annotation `AOBS` of new observation type in loop `1003`.

(6) *Subroutine OBSERVAT*:
In this subroutine, the system response calculated by the TOUGH2 code is compared to the corresponding observation. Assign model results (usually standard TOUGH2 variables or a function thereof) to variable `XTOUGH` in loop `1003` (see examples therein). Make sure that the TOUGH2 variable is available either by a standard TOUGH2 common block or by a newly created common block which transfers values calculated by TOUGH2 (usually in subroutine *MULTI* or *EOS*) to subroutine *OBSERVAT*.

It is strongly suggested to update the version control statements at the beginning of each updated subroutine. Furthermore, the syntax of the new command should be documented and sent to the author of this manual together with a listing of the updated subroutine `OBSERVAT` to make it available for other users.

Appendix A3: Adding new TOUGH2 modules

The purpose of this Appendix is to help a user add newly developed TOUGH2 modules (e.g. a new EOS module) to the ITOUGH2 code. The main idea leading to the current architecture of ITOUGH2 is described in Section 4.2 and visualized in Figure 8. ITOUGH2 is designed for easy enhancing the code's capabilities to solve both the direct and the inverse problem. However, a few changes have to be made in order to make standard TOUGH2 modules compatible with ITOUGH2. Here they are:

(1) Replace "`READ IFORMAT`" by "`READ (5,IFORMAT)`"

(2) Replace "`WRITE IFORMAT`" by "`WRITE (6,IFORMAT)`"

(3) Note that file t2m.f which contains the TOUGH2 main program is not linked to the ITOUGH2 executable. `PROGRAM TOUGH2` is replaced by `PROGRAM ITOUGH2` in file it2MAIN.f. Consequently, all major arrays (including those of the original TOUGH2 code) are now dimensioned in the main program ITOUGH2. If new common blocks are introduced, they also should be dimensioned therein. Initialization of common block variables are done in `BLOCK DATA IT`. The original subroutine `IO` which opens all TOUGH2 disk files is replaced by subroutine `OPENFILE`. Note that new names have been assigned to the standard TOUGH2 files. Subroutine `CYCIT` is no longer called by ITOUGH2. Instead, subroutine `CALLTOUG` performs time stepping. In summary: any changes or new developments affecting or replacing the TOUGH2 main program or subroutines `IO` or `CYCIT` are to be accordingly made in the ITOUGH2 main program and in subroutines `OPENFILE` and `CALLTOUG`, respectively.

(4) If an EOS module is added, provide a common block and DATA statement in file eos#.f as follows:

```
COMMON/EOSID/IDEOS
DATA IDEOS/#/
```

where # is the number of the EOS module. Provide information about the primary variables in subroutine `ITHEADER` through variable `CEOS`.

Appendix A4: Customize ITOUGH2

This Appendix helps a user customize his or her ITOUGH2 version.

Default format of plot files:
ITOUGH2 contains several interfaces to reformat the original PLOPO plotfile for different visualization software. Set the default format in `BLOCK DATA IT` through variable `IPLOTFMT` as follows:

`IPLOTFMT = 1` :	PLOPO (default)
`IPLOTFMT = 3` :	AVS Graph Viewer
`IPLOTFMT = 4` :	Columns x y_1 y_2 y_3 ...
`IPLOTFMT = 5` :	IGOR Macintosh, Macro file preceding data
`IPLOTFMT = 6` :	TECPLOT XY-Plot

Additional interfaces can be written following the instructions given in Section 4.3.5.5 under command `>>> PLOTFILE`.

Machine dependent subroutines:
While ITOUGH2 is written in standard FORTRAN 77, some machine dependent functions are used (mainly date and time information). They are provided in files mdepIBM.f, mdepSTAR.f, mdepSUN.f, and mdepCRAY.f for IBM workstations, STARDENT mini-supercomputers, SUN workstations, and CRAY supercomputers, respectively. If a different operating system is used, copy one of the files mentioned above and replace the machine dependent subroutines and functions. Modify the makefile (Figure 6) accordingly (see Section 4.1).

Using libraries:
A user may wish to employ his or her own subroutines for optimization, matrix operations etc. commonly provided by commercial software packages. Copy file it2XXXX.f or use file it2ULIB.f to write the appropriate interfaces. Modify the makefile (Figure 6) accordingly (see Section 4.1).

Changing commands or keywords:
If you wish to change the keywords, modify them or provide alternatives in the DATA-statements preceding each subroutine in file it2INPUT.f which reads input. Avoid redundancy on the same command level.

Help file
Provide full path of help file <*itough2.help*> through variable `CHELP` in `BLOCK DATA IT`.

Appendix A5: Subroutines and Functions

Subroutines (S) and functions (F) of the ITOUGH2 code (excluding those of the standard TOUGH2 simulator) are listed in alphabetic order.

NAME	S/F	FILE	NAME	S/F	FILE
ACTOF	F	it2XXXX.f	HESSE	S	it2MAIN.f
AMULTB	S	it2IMSL.f	INVERT	S	it2IMSL.f
ANNEAL	S	it2MAIN.f	INANNEAL	S	it2INPUT.f
ATMULTA	S	it2IMSL.f	INCOMPUT	S	it2INPUT.f
ATMULTB	S	it2IMSL.f	INCOVARI	S	it2INPUT.f
BESTRUN	S	it2MAIN.f	INELEM	S	it2INPUT.f
BREAKHA	F	mdepIBM.f	INERROR	S	it2INPUT.f
BRENT	F	it2XXXX.f	INIGUESS	S	it2MAIN.f
CALLTOUG	S	it2MAIN.f	ININIGUE	S	it2INPUT.f
CONJUG	S	it2IMSL.f	INITTOUG	S	it2MAIN.f
CPUSEC	S	mdepIBM.f	INJACOB	S	it2INPUT.f
DAYTIM	S	mdepIBM.f	INOBS	S	it2INPUT.f
DETERMI	S	it2IMSL.f	INOBSDAT	S	it2INPUT.f
DIRSEA	S	it2IMSL.f	INOBSERV	S	it2INPUT.f
EIGEN	S	it2MAIN.f	INOPTION	S	it2INPUT.f
ERROR	S	it2INPUT.f	INPAIRED	S	it2INPUT.f
EVALOBJF	S	it2MAIN.f	INPAR	S	it2INPUT.f
EVALVEC	S	it2IMSL.f	INPARAME	S	it2INPUT.f
EXRACT	S	it2IMSL.f	INPOLY	S	it2INPUT.f
FCNLEV	S	it2MAIN.f	INPRINT	S	it2INPUT.f
FCNNEWT	S	it2MAIN.f	INQXX	S	it2INPUT.f
FINDKEY	S	it2INPUT.f	INTIMES	S	it2INPUT.f
FLOPP	S	it2MAIN.f	INTOLER	S	it2INPUT.f
F1DIM	F	it2XXXX.f	INWBP	S	it2INPUT.f
GASDEV	F	it2MAIN.f	IN_OUT	S	it2INPUT.f
GETINCON	S	it2MAIN.f	ITHEADER	S	it2INPUT.f
GETINDEX	S	it2MAIN.f	ITINPUT	S	it2INPUT.f
GETMESH	S	it2MAIN.f	INWEIGHT	S	it2INPUT.f
GETNMAT	S	it2MAIN.f	IXLBXUB	S	it2MAIN.f

NAME	S/F	FILE
GRADNEW	S	it2MAIN.f
JAC	S	it2MAIN.f
LENOS	S	it2INPUT.f
LEVMAR	S	it2IMSL.f
LINESEA	S	it2MAIN.f
LOGLIKE	S	it2MAIN.f
LTU	S	it2INPUT.f
LUFACT	S	it2IMSL.f
MEAN	S	it2MAIN.f
MLBUB	S	it2MAIN.f
MLLAMBDA	S	it2MAIN.f
MMNN	S	it2MAIN.f
MTCARLO	S	it2MAIN.f
NEXTWORD	S	it2INPUT.f
NONLIN	S	it2MAIN.f
OBJFUN	S	it2MAIN.f
OBSERVAT	S	it2MAIN.f
OBSERVED	F	it2MAIN.f
OBSMEAN	S	it2MAIN.f
OPENFILE	S	it2INPUT.f
PERFIND	F	it2IMSL.f
PLOTCHAR	S	it2MAIN.f
PLOTFILE	S	it2MAIN.f
PLOTIF	S	it2MAIN.f
POLYNOM	F	it2MAIN.f
PREC	S	it2MAIN.f
PRINTML	S	it2IMSL.f
PRIORINF	S	it2MAIN.f
PRSTATUS	S	it2INPUT.f
QCHI	F	it2MAIN.f
QFISHER	F	it2MAIN.f
QNEWTON	S	it2IMSL.f
QNORMAL	F	it2MAIN.f
QUOTES	S	it2MAIN.f
RANDOM	F	it2MAIN.f
READCOMM	S	it2INPUT.f
READINT	S	it2INPUT.f
READREAL	S	it2INPUT.f
REFORMAT	S	it2MAIN.f
RESIDUAL	F	it2MAIN.f
SECOND	S	mdepIBM.f
SETIO	S	it2INPUT.f
SETIR	S	it2IMSL.f
SETIR2	S	it2IMSL.f
SETPLL	S	it2MAIN.f
SETWSCAL	S	it2MAIN.f
SETXSCAL	S	it2MAIN.f
SKIP	S	it2MAIN.f
TERMINAT	S	it2MAIN.f
THEADER	S	it2INPUT.f
TIMEWIND	S	it2MAIN.f
TRANSA	S	it2IMSL.f
UPDATE	S	it2MAIN.f
USERBC	S	it2USER.f
USERFUNC	S	it2USER.f
USEROBS	S	it2USER.f
USERPAR	S	it2USER.f
VARIANCE	S	it2MAIN.f
WHATCOM	S	mdepIBM.f
WHATLIB	S	it2IMSL.f

Appendix A6: Updates

V1.1: December 4, 1992

New subroutines:	INUSER, INUSROBS, USERPAR, USEROBS, PLOTCHAR, PLOTIF, REFORMAT, QUOTES
Modified subroutine:	INPAIRED, INERROR, MTCARLO, INPRINT, IN_OUT, TERMINAT, INELEM, UPDATE, OBSERVAT, OPENFILE, THEADER, FINDKEY, GETMESH, INOPTION, INWEIGHT, LINEQ, RESIDUAL, TERMINAT, INFLOW, INOBSDAT
New commands:	CHARACTERISTIC, CLASS, FISHER, FORMAT, ROBUST ESTIMATOR 1/2
New keywords:	BRINE, DAY, GAS, HOUR, LIQUID, MINUTE, MONTH, NAPL, SECOND, WEEK, YEAR
New Features:	- Allows to specify time units for paired data sets - Number of classes can be specified for histogram drawing - User specified parameter and observation types added - Plots characteristic curves - Interface to plotting programs, reformatting of PLOPO plotfile - Fisher Model Test

V2.0: August 1, 1993

New subroutines:	INANNEAL, ANNEAL
New commands:	AFTER, ANNEAL, BEFORE, ITERATION, SCHEDULE, STEP, TEMPERATURE
New Features:	- Simulated Annealing minimization

V2.1: September 29, 1993

New subroutines:	INPAR, INOBS
Deleted subroutines:	INABSPER, INRELPER, INCAPPRE, INCOMPRE, INPOROSI, ININCON, INBOUPRE, INUSER, INPRESSU, INFLOW, INTEMP, INUSROBS, JVLF
Modified subroutines:	IN_OUT, INPARAME, ININGUE, INWBP, INOBSERV, INOBSDAT, INELEM, PRSTATUS, ERROR, USERPAR, USEROBS, ITOUGH2, MMNN, UPDATE, INIGUESS,

	SETWSCAL, IXLBXUB, GETNMAT, GETINDEX, GETINCON, PRIORINF, OBJFUN, LOGLIKE, MLLAMBDA, ANNEAL, LINESEA, JAC, HESSE, OBSERVAT, TIMEWIND, BESTRUN, EVALOBJF, MTCARLO, PLOTFILE, EIGEN, NONLIN, TERMINAT
New commands:	ANNOTATION, AVERAGE, GENERATION, MEAN, MINC, SELEC, SKIN, SUM
Deleted commands:	LOCATION
New Features:	- Parameter and observation vector rearranged for easy enhancement of the code in the future - Read annotation of parameter and observation - Estimate skin radius, constant generation rate, MINC parameters and SELEC parameters - Allow for multiple definition of material names and grid blocks - Compute sum or mean value of state variables, if more than one grid block or connection is given

V2.2: February 1, 1994

Modified subroutines:	IN_OUT, INOBSERV, INOBS, INTOLER, INOBSDAT, INPAIRED, UPDATE, MLLAMBDA, OBSERVAT, BALLA, EVALOBJ, TERMINAT, USEROBS, TERMINAT
New commands:	COMMAND INDEX, COMPONENT, CONCENTRATION, INPUT, MASS, PHASE, REFERENCES, STEADY-STATE, SUBROUTINES, VOLUME
New keywords:	HELP, LIST
New Features:	- New parameters: . Total mass of components or phases . Total volumes of phases . Mass fractions (concentrations) - New observation type: . Flowing enthalpy - Flexible last time point for steady-state calculations - Help module - Prints command index, references, updates, and list of subroutines

Appendix A7: Command Index