LBNL-40041 Rev.

iTOUGH2 V7.0
Command Reference

Stefan Finsterle

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

May 2015

This work was supported by the U.S. Dept. of Energy under Contract No. DE-AC02-05CH11231.

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor The Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or The Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, or The Regents of the University of California.

Ernest Orlando Lawrence Berkeley National Laboratory
is an equal opportunity employer.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PAGE INTENTIONALLY LEFT BLANK

1. INTRODUCTION

This report contains a detailed description of all iTOUGH2 commands. It complements the "iTOUGH2 User's Guide" [*Finsterle*, 2007a], and the collection of iTOUGH2 sample problems [*Finsterle*, 2007c].

iTOUGH2 is a program for parameter estimation, sensitivity analysis, and uncertainty propagation analysis. It is based on the TOUGH2 simulator for non-isothermal multiphase flow in fractured and porous media [*Pruess*, 1987, 1991a]. Extensive experience in using TOUGH2 is a prerequisite for using iTOUGH2. The preparation of an input file for TOUGH2 or its derivatives is described in separate manuals and is not part of this report.

The "iTOUGH2 User's Guide" [*Finsterle*, 2007a] summarizes the inverse modeling theory pertaining to iTOUGH2, and describes the program output. Furthermore, information about code architecture and installation are given. In Section 2 of this report, a brief summary of inverse modeling theory is given to restate the main concepts implemented in iTOUGH2 and to provide certain definitions. Section 3 introduces the basic concepts of the iTOUGH2 input language and the main structure of an iTOUGH2 input file. Chapter 4, the main part of this report, provides detailed descriptions of each iTOUGH2 command in alphabetical order. It is complemented by a command index in Appendix A in which the commands are given in logical order. The content of Section 4 is also available on-line at http://esd.lbl.gov/iTOUGH2 (click on "Command Index"). Section 5 describes the usage of the UNIX script files for executing, checking, and terminating iTOUGH2 simulations.

A variety of inverse problems solved by iTOUGH2 are discussed in a collection of sample problems [*Finsterle*, 2007c], which includes a tutorial example illustrating the main features of iTOUGH2. Complete examples of iTOUGH2 input files are given along with interpretations of the corresponding output files.

The key to a successful application of iTOUGH2 is (i) a good understanding of multiphase flow processes, (ii) the ability to conceptualize the given flow and transport problem and to develop a corresponding TOUGH2 model, (iii) detailed knowledge about the data used for calibration, (iv) an understanding of parameter estimation theory and the correct interpretation of inverse modeling results, (v) proficiency in using iTOUGH2 options. This report addresses issue (v) only.

2. iTOUGH2 SUMMARY DESCRIPTION

A comprehensive description of inverse modeling theory implemented in iTOUGH2 is given in *Finsterle* [2007a]. The purpose of the summary description provided here is mainly to introduce basic concepts and notations referred to in Section 4 of this report.

iTOUGH2 is a computer program that provides inverse modeling capabilities for the TOUGH2 code. While the main purpose of iTOUGH2 is to estimate model-related hydraulic properties by calibrating TOUGH2 models to laboratory or field data, the information obtained by evaluating parameter sensitivities can be used to scrutinize the design of an experiment and to evaluate the uncertainty of model predictions.

iTOUGH2 solves the inverse problem by automatic model calibration. All TOUGH2 input parameters can be considered unknown or uncertain. The parameters are estimated based on any type of observation for which a corresponding TOUGH2 output is available, including prior information about the parameters to be estimated. A number of different objective functions and minimization algorithms are available. One of the key features of iTOUGH2 is its extensive error analysis, which provides statistical information about residuals, estimation uncertainties, and the ability to discriminate among model alternatives. The impact of parameter uncertainties on model predictions can be studied by means of linear uncertainty propagation analysis or Monte Carlo simulations.

The key elements of an inverse modeling code are (1) a simulation program to model flow and transport in the hydrogeologic system ("forward modeling"), (2) the objective function which measures the misfit between the model output and the data, (3) the minimization algorithm which reduces the objective function by automatically updating parameter values, and (4) the error analysis which allows one to judge the quality of the estimates.

iTOUGH2 estimates elements of a parameter vector $\mathbf{p}$ based on observations summarized in vector $\mathbf{z}^*$ by minimizing an objective function S which is a function of the residual vector $\mathbf{r}(\mathbf{p})$.

Vector $\mathbf{p}$ of length n contains the parameters to be estimated by inverse modeling, or the parameters considered uncertain for uncertainty propagation analysis.

Parameter vector: $$\mathbf{p}^T = [p_1, p_2, \ldots, p_n] \qquad \text{(Eq. 1)}$$

In the simplest case, the parameters p_i to be estimated are identical with certain TOUGH2 input parameters. For example, p_1 is the porosity of a certain rock type, and p_2 is a parameter of the capillary pressure function. Other potential parameters include initial and boundary conditions, or geometrical features (such as fracture spacing). iTOUGH2 also allows one to estimate a single parameter that will be assigned to multiple TOUGH2 input

variables, i.e., p_i can represent more than one TOUGH2 variable. Moreover, user-specified parameters can be programmed into iTOUGH2.

Simple parameter transformations can be employed to make the inverse problem more linear or to change the distributional assumption about the parameter. For example, the log-normal character of permeability suggests estimating the logarithm of permeability rather than permeability itself. While the transformation does not change the best estimate parameter set, it makes the detection of the corresponding minimum of the objective function more robust. Different results are obtained, however, when prior information is incorporated or when uncertainty propagation analyses are performed. In these cases a prior standard deviation must be given which represents either a normal or log-normal distribution, respectively.

Table 1 shows the parameter transformations available in iTOUGH2. Here, X is one or more TOUGH2 input variables (with initial guess X_0), and p is the parameter estimated by iTOUGH2. It is important to realize that the standard deviation, parameter variation, acceptable parameter range, and the starting point for the optimization refer to p (i.e., not to X).

Table 1. Parameter Transformations

$X \rightarrow p$	$p \rightarrow X$	iTOUGH2 command
$p = X$	$X = p$	`>>>> VALUE`
$p = \log_{10}(X)$	$X = 10^p$	`>>>> LOGARITHM`
$p = X/X_0$	$X = p \cdot X_0$	`>>>> FACTOR`
$p = \log_{10}(X/X_0)$	$X = 10^p \cdot X_0$	`>>>> LOG(F)`

Vector $\mathbf{z}$ of length m contains dependent, observable variables, usually related to measurements taken at discrete points in space $\mathbf{x}$ and time t. Such a selected point in space and time, $z_i(\mathbf{x}_j, t_k)$, will be referred to as a calibration point. Elements of $\mathbf{z}$ refer to both measured quantities (data)—indicated by an asterisk (*)—and simulation results. The most commonly used observations for calibration are pressure, flow rate, temperature and concentration measurements.

$$\text{Observation vector:} \quad \mathbf{z}^T = \mathbf{z}(\mathbf{x}_j, t_k)^T = [z_1, \ldots, z_n, z_{n+1}, \ldots z_m] \qquad \text{(Eq. 2)}$$

The vector of observable variables may also contain measured parameter values. For example, if permeability has been measured on cores in the laboratory, this information can be considered as an additional data point, and treated along with the direct observations of the system response. Such measured parameter values are referred to as "prior information." The first n elements of $\mathbf{z}$ are therefore identical to the variables of $\mathbf{p}$.

The residual vector $\mathbf{r}$ contains the differences between the measured and calculated system response; the latter is a function of parameter vector $\mathbf{p}$:

$$\text{Residual vector:} \qquad \mathbf{r}^T = (\mathbf{z}^* - \mathbf{z}(\mathbf{p}))^T = [r_1, \ldots, r_n, r_{n+1}, \ldots, r_m] \qquad \text{(Eq. 3)}$$

For example, element $r_i (i > n)$ is the difference between the measured and calculated pressure at a certain point in space and time. A special type of residual (for $i \le n$) is the difference between the measured parameter p_i^* (prior information, if available) and the estimated parameter value.

The elements of vectors $\mathbf{z}$ and $\mathbf{r}$ may have different physical meanings with different units of measurement, and represent observations of different accuracy. The inverse of the covariance matrix of $\mathbf{z}$ can be used as a scaling matrix. If taking the view of maximum likelihood estimation, the covariance matrix is part of the stochastic model, i.e., it represents the distributional assumption about the final residuals. In the absence of modeling errors, the residuals are equal to the measurement errors, and therefore the covariance matrix is often based on the error structure of the data. A reasonable assumption about the measurement errors is that they are a result of many individual error sources, and are thus uncorrelated, normally distributed random variables with zero mean. The distributional assumption can therefore be summarized in a covariance matrix $\mathbf{C}_{zz}$, which is an $m \times m$ diagonal matrix. The i-th diagonal element of matrix $\mathbf{C}_{zz}$ is the variance σ_i^2 representing the uncertainty of the final residual (or measurement error) of observation z_i:

$$\mathbf{C}_{zz} = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_i^2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_n^2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \sigma_j^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \sigma_m^2 \end{bmatrix} \qquad \text{(Eq. 4)}$$

The first n diagonal elements are the variances of the prior information vector $\mathbf{p}^*$, followed by $m - n$ variances of the observed system state. Note that only the relative magnitude of the elements of $\mathbf{C}_{zz}$ influences the values of the estimated parameters. We therefore introduce a dimensionless factor σ_0^2, which is termed the prior error variance, and a positive definite matrix $\mathbf{V}_{zz}$, where $\mathbf{V}_{zz}^{-1}$ will be used as a weighting matrix:

$$\mathbf{C}_{zz} = \sigma_0^2 \cdot \mathbf{V}_{zz} \qquad \text{(Eq. 5)}$$

While σ_0^2 can assume any positive number, it is convenient to set σ_0^2 = 1, i.e., the weighting matrix is the inverse of the covariance matrix.

After the inversion, the *a posteriori* or *estimated error variance* s_0^2 is calculated (see Eq. 10 below). If the preconception about the final residuals is correct, and if the true system response is identified, the ratio s_0^2 / σ_0^2 should not significantly deviate from 1.0.

The objective function S is a measure of the misfit between the data and the model calculation. If the residuals $(\mathbf{z}^* - \mathbf{z})$ are normally distributed with mean $E[(\mathbf{z}^* - \mathbf{z})] = \mathbf{0}$ and covariance matrix $E[(\mathbf{z}^* - \mathbf{z})(\mathbf{z}^* - \mathbf{z})^T] = \mathbf{C}_{zz}$, maximum-likelihood theory yields the weighted least-squares objective function:

$$S = (\mathbf{z}^* - \mathbf{z})^T \mathbf{C}_{zz}^{-1} (\mathbf{z}^* - \mathbf{z}) = \mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r} = \sum_{i=1}^{m} \frac{r_i^2}{\sigma_i^2} \qquad \text{(Eq. 6)}$$

Alternative objective functions are available in iTOUGH2 (see *Finsterle* [2007a]). The best estimate parameter set minimizes the objective function (Eq. 6). Minimization of the objective function S is based on local linearization (see Levenberg-Marquardt algorithm below). The partial derivatives of the calculated system response with respect to the parameters are summarized in the Jacobian matrix $\mathbf{J}$ of dimensions $m \times n$, the elements of which are defined as follows:

$$J_{ij} = -\frac{\partial r_i}{\partial p_j} = \frac{\partial z_i}{\partial p_j} \qquad \text{(Eq. 7)}$$

iTOUGH2 calculates $\mathbf{J}$ numerically by means of either forward or centered finite differences. The Jacobian is also important in the *a posteriori*, linear error analysis as well as the uncertainty propagation analysis.

While iTOUGH2 offers a number of different minimization algorithms, the Levenberg-Marquardt modification of the Gauss-Newton algorithm is found to be a rather general and robust procedure for iteratively updating parameter vector $\mathbf{p}$ [*Levenberg*, 1944; *Marquardt*, 1963]. It can be described as shown in Table 2. If λ is large, the first term on the right-hand side of Eq. (8) becomes a matrix with a dominant diagonal. This leads to a small step along the gradient of *S*. Stepping along the steepest descent direction is robust, but inefficient. The Levenberg parameter λ is decreased after each successful step. With decreasing λ, the parameter update $\Delta\mathbf{p}$ (Eq. 9) converges to the one proposed by the Gauss-Newton algorithm with its quadratic convergence rate.

Table 2. Levenberg-Marquardt Minimization Algorithm

Step 1:	Initialization: - Set iteration index $k = 0$. - Define initial Levenberg parameter (default: $\lambda_0 = 10^{-3}$). - Define Marquardt parameter (default: $\nu = 10$). - Define initial parameter set $\mathbf{p}_{k=0} = \mathbf{p}_0$.
Step 2:	Run simulation model with parameter vector $\mathbf{p}_k$.
Step 3:	Evaluate $\mathbf{r}(\mathbf{p}_k)$, $S(\mathbf{p}_k)$, and $\mathbf{J}(\mathbf{p}_k)$.
Step 4:	Calculate parameter update: $\Delta\mathbf{p}_k = \left(\mathbf{J}_k^T\mathbf{C}_{zz}^{-1}\mathbf{J}_k + \lambda_k\mathbf{D}_k\right)^{-1}\mathbf{J}_k^T\mathbf{C}_{zz}^{-1}\mathbf{r}_k$ (Eq. 8) where $\mathbf{D}_k$ is an $n \times n$ Tikhonov matrix
Step 5:	Update parameter vector: $\mathbf{p}_{k+1} = \mathbf{p}_k + \Delta\mathbf{p}_k$. (Eq. 9)
Step 6:	Perform simulation and evaluate $S(\mathbf{p}_{k+1})$.
Step 7:	If $S(\mathbf{p}_{k+1}) < S(\mathbf{p}_k)$ multiply λ by factor $1/\nu$ and go to Step 8. If $S(\mathbf{p}_{k+1}) > S(\mathbf{p}_k)$ multiply λ by factor ν and go to Step 4.
Step 8:	Evaluate convergence criteria. If converged, go to Step 9, else set $k = k + 1$ and go to Step 2.
Step 9:	Minimum identified. Proceed with residual and uncertainty analysis.

One of the key advantages of a formalized approach to parameter estimation is the possibility to perform an extensive *a posteriori* error analysis. First, the residual analysis provides some measure of the overall goodness-of-fit and allows identification of systematic errors or flaws in the stochastic model. Next we can determine the uncertainty of the estimated parameters. Note that a good match does not necessarily mean that the estimates are reasonable. They may be highly uncertain due to a lack of sensitivity or high parameter correlations, which is an indication that the inverse problem is over-parameterized. The covariance matrix of the estimated parameters can be further analyzed to obtain correlation coefficients, indicating parameter combinations that lead to similar matches, etc. Finally, we can calculate the uncertainty of the simulation results, which also allows us to identify potential outliers in the data.

The estimated error variance s_0^2 represents the variance of the weighted residuals and is thus a measure of goodness-of-fit:

$$s_0^2 = \frac{\mathbf{r}^T \mathbf{V}_{zz}^{-1} \mathbf{r}}{m - n} \qquad \text{(Eq. 10)}$$

Note that if the residuals are consistent with the distributional assumption about the measurement errors (i.e., matrix $\mathbf{C}_{zz}$), then the estimated error variance assumes a value close to σ_0^2. Next we calculate the expected value and the covariance matrix $\mathbf{C}_{pp}$ of the estimated parameters. Based on the linearity assumption it is easy to show that the expected value of the estimated parameter is equal to the parameter itself. The definition of the covariance matrix yields:

$$\mathbf{C}_{pp} = s_0^2 \left(\mathbf{J}^T \mathbf{V}_{zz}^{-1} \mathbf{J} \right)^{-1} \qquad \text{(Eq. 11)}$$

The interpretation of the covariance matrix $\mathbf{C}_{pp}$ provides the key criteria to evaluate the inverse modeling results.

This concludes the summary description of iTOUGH2. Again, more details about inverse modeling theory, objective functions, minimization algorithms, and the residual and error analysis can be found in *Finsterle* [2007a].

3. BASIC CONCEPTS

Formats of iTOUGH2 Input Files

The user must provide at least two input files to run iTOUGH2. The first one is a TOUGH2 input file in standard TOUGH2 format as described in *Pruess* [1987, 1991a] and *Falta et al.* [1995], as well as other publications pertaining to particular TOUGH2 modules and code enhancements (e.g., *Pruess* [1991b], *Finsterle et al.* [1994], *Moridis and Pruess* [1995]). This input file defines the conceptual model. It solves the forward problem and should run successfully using standard TOUGH2 not only for the initial parameter set, but also for a wide range of potential parameter combinations.

The second input file is the iTOUGH2 input file, in which the user specifies the parameters to be estimated, the observations used for calibration, and various program options (see Section 3.3). A special command interpreter has been developed to read the iTOUGH2 input file. The basic concept of the iTOUGH2 input language is discussed in Section 3.2. A detailed description of each iTOUGH2 command is given in Chapter 4.

Basic Concepts of iTOUGH2 Input Language

Execution of iTOUGH2 is controlled by means of a high-level input language. The commands are hierarchically structured, i.e., each command has a parent command on a higher command level, and potentially one or more child commands on a lower command level. The command level is identified by the corresponding number of ">" markers (e.g., ">>>" to enter command level three). Each command level must be terminated by reversed markers (e.g., "<<<" to quit command level three and return to level two). The position of the command level marker on a line is irrelevant.

A line *without* a command level marker is considered a comment and is skipped, unless it contains data requested by a previous command. Entire sections of the input file may be commented out by marking the first and last line with slash-star (`/*`) and star-slash (`*/`), respectively. All lines between these two marks are not interpreted even if they contain command level markers. The number sign (#) in the first column comments out the respective line. The double-slash (`//`) tells iTOUGH2 to ignore everything that follows the slashes until the end of the line.

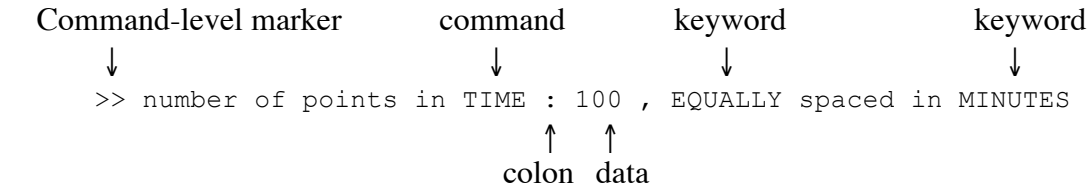
On the line containing a command level marker, a command is expected which is unique to the corresponding command level and the associated parent command. A command can be complemented by optional keywords. Additional text present on the command line is ignored. Commands and keywords are case-insensitive. The position of the command and keywords on a command line is arbitrary. Many commands request additional numerical or alphanumeric input. This input can be provided anywhere on the command line, but must immediately follow a colon (`:`). Input may consist of one or more integers, real numbers,

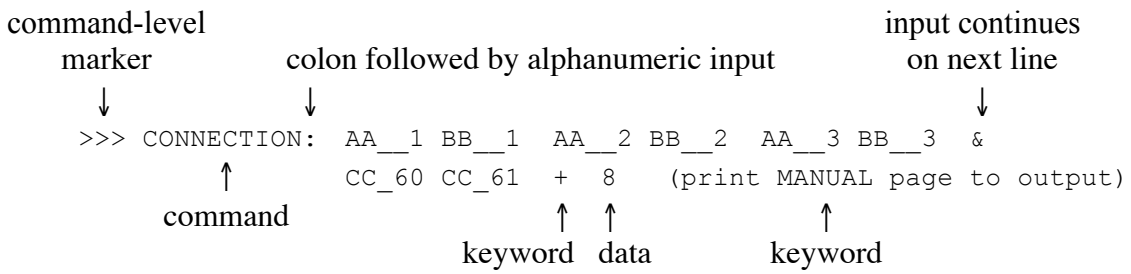
or strings. Input belonging to the same command can be continued on following lines by using the "&" character. Each line of the input file is processed word by word, the delimiter being one or more spaces. All numerical data are read in free format. Long lists of data start one line after the command line. Data lists will be read until a FORTRAN reading error occurs.

Command `LIST` can be used on any command level to obtain a list of acceptable commands on the current command level. The keyword `HELP` or `MANUAL` can be used on any command line to print a short tutorial of the corresponding command to the output file.

In general, subcommands of the same parent command can be provided in arbitrary order. A few exceptions apply; they are clearly identified when using an incorrect order.

Examples:





Main Structure of iTOUGH2 Input

There are three main blocks in the iTOUGH2 input file, identified by one of the following first-level commands:

```
> PARAMETER
> OBSERVATION
> COMPUTATION
```

The first block (first-level command `> PARAMETER`) is used to identify those TOUGH2 input parameters that will be subjected to parameter estimation, sensitivity analysis, or uncertainty propagation analysis. A series of subcommands ensure a unique identification of the parameter by providing its type (second-level commands), and the domain it is refer-

ring to (third-level commands). The fourth-level commands are used to provide further specifications along with statistical details about the parameter.

The second block (first-level command `> OBSERVATION`) is used to identify those TOUGH2 output variables that will be compared to observed data for model calibration. Furthermore, the points in time at which calibration should occur are also specified in this block. An observable variable is identified by its type (second-level command), its location (third-level command), as well as more detailed specifications if necessary (fourth-level commands). The measured data to which the calculated system response is compared to is also provided through a fourth-level command.

The third block (first-level command `> COMPUTATION`) is used for various program options and computational parameters. It deals with convergence criteria, the numerical calculation of the Jacobian matrix, statistical parameters for the error analysis, and output options. Most important, the application mode of iTOUGH2 (parameter estimation, sensitivity analysis, uncertainty propagation analysis) can be selected along with a number of additional features.

Figure 1 depicts the main structure of an iTOUGH2 input file in a generic way. A list of available subcommands can be found in Appendix A, and a detailed description of each iTOUGH2 command is given in Section 4.

```
> PARAMETER
  >> specify parameter type
     >>> specify parameter domain
         >>>> provide details
         <<<<
     <<<
  <<

> OBSERVATION
  >> specify calibration points in TIME
  >> specify observation type
     >>> specify location
         >>>> provide details
         >>>> provide data
         <<<<
     <<<
  <<

> COMPUTATION
  >> specify various program OPTIONS
  >> specify CONVERGEnce criteria
  >> specify parameters for calculating JACOBIAN matrix
  >> specify parameters for ERROR analysis
  >> specify OUTPUT formats
  <<
```

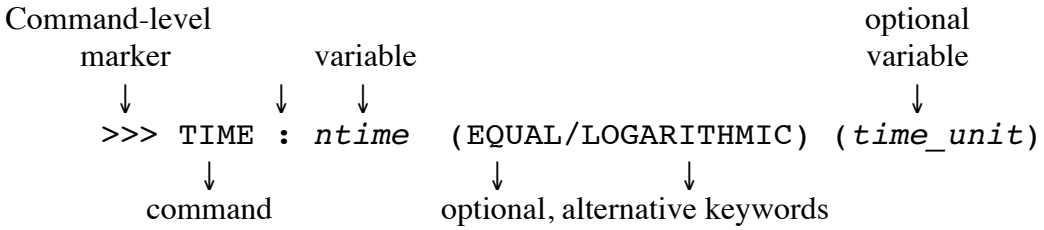
Figure 1. Main structure of iTOUGH2 input.

4. iTOUGH2 COMMANDS

A detailed description of each iTOUGH2 command is given below. The description includes the command syntax, the name of the parent command, the name of subcommands (if applicable), as well as the purpose and effect of the command. Theoretical background is provided where necessary. Furthermore, an illustrative example is given for each command.

In general, the command syntax includes the command-level marker, the command name as identified by iTOUGH2, input variables (in *italics*), and optional keywords or variables (in parentheses). Alternatives are separated by a slash (/).

Example:



In this command reference, the commands are given in alphabetic order. Note that the name of a command is not always unique, in which case the command level and the parent command can be used for identification. If reference is made to a command with a non-unique command name, the first character of the parent command (or the corresponding first-level command) is indicated in parentheses (e.g., `>>> FORWARD (c)` refers to the command in block `>> CONVERGE`, whereas `>>> FORWARD (j)` refers to the command in block `>> JACOBIAN`).

iTOUGH2 commands can be written in lower or upper case. In this manual, the commands and keywords are always typed in upper case for easy identification. Therefore, the lower case words on the command line are comments that will be ignored by the iTOUGH2 command interpreter.

The examples given for each command usually illustrate the basic usage of a command. In many cases, however, extended features and advanced applications are provided to demonstrate the use of a command in combination with other commands. The examples are not complete and can therefore not be run separately. Furthermore, they may have to be modified to avoid warning messages.

The command description can also be accessed on-line at http://esd.lbl.gov/iTOUGH2 (click on `Command Index`).

Command
```
>>> ABORT (NO)
```

Parent Command
```
>> STOP/CONVERGE
```

Subcommand
-

Description
It may be appropriate to abort a simulation if the value of the objective function (as accumulated to the current simulation time) exceeds the minimum objective function value, indicating that the run will not lead to a reduction in the objective function. (Note that aborting such simulations is not appropriate when evaluating the Jacobian matrix, or when using a global minimization algorithm.)

This option may reduce overall CPU time. It is the default option when evaluating trial steps in the Levenberg-Marquardt algorithm (the default can be overwritten by using command `>>> NO ABORT`).

It may be specifically effective if `>>> GRID SEARCH` is solely used to find the minimum (rather than mapping the entire response surface), or if `>>> MONTE CARLO` and `>>> LATIN HYPERCUBE` are used not for uncertainty propagation analysis, but as a random search algorithm.

Example
```
> COMPUTATION
  >> STOP
     >>> allow runs to ABORT as soon as OF>OFmin
     <<<
  <<
```

See Also
`>>> GRID SEARCH`, `>>> MONTE CARLO`, `>>> LATIN HYPERCUBE`

Command
`>> ABSOLUTE`

Parent Command
`> PARAMETER`

Subcommand
`>>> MATERIAL`

Description
This command selects as a parameter the absolute permeability (TOUGH2 variable *PER(ISOT,NMAT)*) of the specified material.
Subcommand `>>>> LOGARITHM` invokes estimation of a single log-value which is assigned to all selected materials; subcommand `>>>> FACTOR` invokes estimation of a common multiplication factor which is applied to all selected permeabilities, thus maintaining the permeability ratios between the materials; subcommand `>>>> LOG(F)` should be used to estimate a log-normally distributed factor with which all the permeabilities are multiplied. By default, the estimate refers to all three flow directions (*ISOT*=1, 2, 3). Subcommand `>>>> INDEX` can be used to select the permeability of a specific flow direction *ISOT*.

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: BOREH  SKIN_
         >>>> LOGARITHM
         <<<<
     >>> MATERIAL: SAND1 SAND2
         >>>> FACTOR
         >>>> INDEX: 1 2  (horizontal permeability)
         <<<<
     >>> MATERIAL: SAND1 SAND2
         >>>> FACTOR
         >>>> INDEX: 3    (vertical permeability)
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>>> ABSOLUTE
```

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
This command takes the absolute value of the calculated system response as the model output to be compared to the observed data.

Example
```
> OBSERVATION
  >> CAPILLARY PRESSURE
     >>> ELEMENT: ELM99
         >>>> take ABSOLUTE value and compare to ...
         >>>> ... positive DATA stored on FILE: pcap.dat
         >>>> the RELATIVE error is: 5 %
         <<<<
      <<<
  <<
```

See Also
-

Command
```
>>> ADJUST
```

Parent Command
```
>> CONVERGE
```

Subcommand
-

Description
The initial time step size is provided through TOUGH2 variable *DELTEN* or *DLT(1)*, usually followed by automatic time stepping (see TOUGH2 variables *MOP(16)* and *REDLT*). The initial time step may be too big (i.e., is automatically reduced by TOUGH2) or too small (i.e., convergence is achieved within one Newton-Raphson iteration), depending on the parameter set supplied by iTOUGH2 during an inversion. This command allows iTOUGH2 to automatically adjust the initial time step size so that convergence is achieved within more than 1 but less than *MOP(16)* Newton-Raphson iterations. Automatic time step adjustment may improve the speed of an inversion, but may also make the inversion unstable if time discretization errors are significant.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> automatically ADJUST initial time step
         (TOUGH2 variable DELTEN)
     <<<
  <<
```

See Also
```
>>> CONSECUTIVE, >>> REDUCTION
```

Command
```
>>> ALPHA: alpha (%)
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
alpha is a probability used for a variety of statistical tests within iTOUGH2. The choice of *alpha* does not affect the estimated parameter set, but *alpha* is used in the residual analysis, the Fisher model test, the analysis of the resulting distribution when performing Monte Carlo simulations, and the width of the error band when performing FOSM uncertainty propagation analysis. *alpha* is expected to assume values between 0.001 and 0.200 (default: 0.01). Instead of the risk α, one can also provide the confidence level (1- α), in which case *alpha* assumes values between 0.8 and 0.999.
Use keyword `%` if *alpha* is given in percent.

Example
```
> COMPUTATION
  >> STOP
     >>> number of TOUGH2 SIMULATIONS: 200
     <<<
  >> ERROR propagation analysis
     >>> MONTE CARLO simulations
     >>> print quantile for risk ALPHA =: 5.0 %
     <<<
  <<
```

See Also
-

Command
```
>>> ANDREWS: c
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects a robust estimator named Andrews. Given this estimator, the objective function to be minimized is the sum of the cosine functions $\gamma(y_i)$, where y_i is the weighted residual:

$$S = \sum_{i=1}^{m} \gamma(y_i)$$

where

$$\gamma(y_i) = \begin{cases} 1 - \cos(y_i / c) & \text{for } |y_i| \le c\pi \\ 2 & \text{for } |y_i| > c\pi \end{cases}$$

with

$$y_i = \frac{r_i}{\sigma_i}$$

This objective function does not correspond to a standard probability density function. It has the general characteristic that the weight given individual residuals first increases with deviation, then decreases to reduce the impact of outliers. The parameter c indicates the deviation at which residuals are considered to be outliers. If the measurement errors happen to be close to a normal distribution with standard deviation σ_i, then the optimal value for the constant c is $c = 2.1$.
Note that this objective function is minimized using the standard Levenberg-Marquardt algorithm, which is designed for a quadratic objective function. Since (1- cos) can be reasonably well approximated by a quadratic function for small y_i, the Levenberg-Marquardt algorithm is usually quite efficient.

Example
```
> COMPUTATION
  >> OPTION
     >>> use robust estimator ANDREWS with a constant c : 1.5
     <<<
  <<
```

See Also
```
>>> CAUCHY, >>> L1-ESTIMATOR, >>> LEAST-SQUARES,
>>> QUADRATIC-LINEAR
```

Command
```
>>> ANNEAL
```

Parent Command
```
>> OPTION
```

Subcommand
```
>>>> ITERATION
>>>> SCHEDULE
>>>> STEP
>>>> TEMPERATURE
```

Description
This command invokes Simulated Annealing to minimize the objective function S.
The following steps are performed by iTOUGH2, controlled by a number of fourth-level commands:

(1) Define the range of possible parameter values using command `>>>> RANGE` in block `> PARAMETER`.

(2) Define an initial value of the control parameter τ using command `>>>> TEMPERATURE`.

(3) iTOUGH2 generates random perturbations $\Delta\mathbf{p}$ of the parameter vector $\mathbf{p}$. The probability density function of the perturbation is either Gaussian or uniform; the initial standard deviations of these distributions are given by command `>>>> DEVIATION (p)`.

(4) The objective function $S(\mathbf{p}_{k+1})$ for the new parameter set $\mathbf{p}_{k+1} = \mathbf{p}_k + \Delta\mathbf{p}$ is evaluated.

(5) If the objective function is decreased (i.e., $\Delta S = S(\mathbf{p}_{k+1}) - S(\mathbf{p}_k) < 0$), the change is retained. If the objective function is increased (i.e., $\Delta S > 0$), the perturbation is accepted with probability $\Pi = \exp(-\Delta S / \tau)$.

(6) After a sufficient number of perturbations have been accepted (see command `>>>> STEP (a)`), τ is lowered according to the annealing schedule (see command `>>>> SCHEDULE`).

(7) Steps (3) through (6) are repeated until the maximum number of temperature reductions (see command `>>>> TRIAL (a)`) is reached.

This scheme of always taking a downhill step and sometimes taking an uphill step with probability Π depending on τ is known as the Metropolis algorithm.
Simulated Annealing may be especially useful for the minimization of a discontinuous cost function in order to optimize operational parameters.

Example

```
> PARAMETER
  >> pumping RATE
     >>> SINK: EXT_1
         >>>> RANGE: -1E-1  -1E-4
         >>>> LOGARITHM
         <<<<
     <<<
  <<

> COMPUTATION
  >> TIME: 1 [YEARS]
     2.0

  >> USER specified cost function: Extraction cost
     >>> SINK: EXT_1
         >>>> NO DATA
         >>>> WEIGHT (=specific costs): 1.0
         <<<<
     <<<
  <<

> COMPUTATION
  >> OPTION
     >>> a cost function is minimized using L1-ESTIMATOR
     >>> perform minimization using Simulated ANNEALing
         >>>> TEMPERATURE : -0.05 (i.e., 5% of initial cost)
         >>>> update after maximum : 100 STEPS
         >>>> annealing SCHEDULE: 0.95
         >>>> Simulated Annealing ITERATIONS: 50
         <<<<
     <<<
  <<
```

See Also

```
>>> GAUSS-NEWTON, >>> GRID SEARCH, >>> LEVENBERG-MARQUARDT,
>>> SIMPLEX, >>>> ITERATION (a), >>>> SCHEDULE,
>>>> STEP (a), >>>> TEMPERATURE
```

Command
>>>> ANNOTATION: *anno*

Parent Command
any third-level command in blocks > PARAMETER and > OBSERVATION

Subcommand
-

Description
A fifteen-character string *anno* can be provided to label parameters and observations in the iTOUGH2 output file. The annotation does not have any function except for making the iTOUGH2 output more readable (exceptions are the user-specified functions; see commands >> USER (p,o). If no annotation is provided, iTOUGH2 internally generates a string, which allows unique identification of the parameter or observation, respectively. The internally generated annotation can be used to check the correctness of the iTOUGH2 input.

Example
```
> PARAMETER
  >> van Genuchten's CAPILLARY pressure function
     >>> ROCK TYPE                : MATRI
         >>>> ANNOTATION          : AIR ENTRY PRESSURE
         >>>> PARAMETER no.       : 2
         <<<<
     >>> ROCK TYPE                : FRAC1
         >>>> PARAMETER no.       : 2
         <<<<
     <<<
  <<
> OBSERVATION
  >> CONCENTRATION of COMPONENT No.: 3 in PHASE No.: 2
     >>> ELEMENT: ELM10
         >>>> NO DATA
         <<<<
     <<<
  <<
```

In the iTOUGH2 output file, the first parameter is referred to as "AIR ENTRY PRESS".
The annotation internally generated for the second parameter reads "CAP.PR.2 FRAC1", where "2" indicates that the second parameter of the capillary pressure ("CAP.PR.") function referring to rock type "FRAC1" is estimated.
The automatically generated annotation for the observation reads "CONC.(3,2)ELM10".

See Also
-

Command
>>>> AUTO (ADD NOISE)

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command provides automatic weighting of observations. The standard deviation calculated by iTOUGH2 is 10 % of the mean of the observed values for a given data set. It is suggested, however, that the assumed measurement error or expected standard deviation of the final residuals be explicitly provided using command >>>> DEVIATION (o). If keyword ADD NOISE is present, Gaussian noise with zero mean and the given standard deviation is added to the data points. This option may be useful if error-free data were synthetically generated.

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: BH__0
         >>>> AUTOmatic weighting
         >>>> DATA are on FILE:  pres.dat
         <<<<
     <<<
  <<
```

See Also
>>>> DEVIATION (o)

Command

```
>>>> AVERAGE (VOLUME)
```

or

```
>>>> MEAN (VOLUME)
```

Parent Command

any third-level command in block `> OBSERVATION`

Subcommand

-

Description

If multiple elements or connections are provided to indicate the location of a measurement point, iTOUGH2 takes the average of all calculated values as the model output to be compared to the data. The user must ensure that the averaging of the quantity is meaningful. If keyword `VOLUME` is present on the command line, the calculated values are weighted by the respective grid block volumes.

Example

```
> OBSERVATION
  >> GAS CONTENT
     >>> ELEMENTS: A1__1 A1__2 A1__3 A1__4 B1__1 B1__2 &
                    C1__1 C1__2 C1__3 C1__4 D1__1 +3
         >>>> ANNOTATION: Ave. Gas Content
         >>>> Take VOLUME AVERAGE
         >>>> NO DATA
         <<<<
     <<<
  <<
```

See Also

```
>>>> SUM
```

Command
```
>>> BENCHMARK
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
A short benchmark calculation is performed, and the CPU time used is compared to that on a reference machine, and a printout is generated with information about approximate relative computer performance.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> perform BENCHMARK calculation
     <<<
```

On output:

```
 Computer is faster than a SUN ULTRA1 by a factor of:   2.1
```

See Also
-

Command
```
>> BOTTOMHOLE PRESSURE
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> SOURCE
```

Description
This command selects as a parameter the bottomhole pressure for wells on deliverability (TOUGH2 variable ***EX***). This parameter refers to a sink/source code name. The generation type must be DELV.

Example
```
> PARAMETER
  >> BOTTOMHOLE PRESSURE in well of deliverability
     >>> SINK: WEL_1 + 5
         >>>> ANNOTATION: wellb. pres. Pwb
         >>>> estimate VALUE
         >>>> RANGE     : 0.5E5 5.0E5 [Pa]
         <<<<
     <<<
  <<
```

See Also
-

Command
`>>>> BOUND: lower upper`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
(synonym for command `>>>> RANGE`)

Example
(see command `>>>> RANGE`)

See Also
`>>>> RANGE`

Command
```
>> BOX-COX
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects parameter λ of the Box-Cox transformation as the parameter to be adjusted or estimated. To address a violation of the homoscedasticity assumption, the Box-Cox transformation [*Box and Cox*, 1964] can be applied to the measured and simulated data as follows:

$$\tilde{z} = \begin{cases} \dfrac{(z+a)^{\lambda} - 1}{\lambda \cdot g^{\lambda - 1}} & \text{if } \lambda \neq 0 \\ g \cdot \ln(z+a) & \text{if } \lambda = 0 \end{cases}$$

Here, g is the geometric mean of the data in the data set, and a is an internally calculated shift parameter, ensuring that the argument to the power function is positive. The Box-Cox transformation relates to a data set in block `>> OBSERVATION`, which must contain the command `>>>> BOX-COX:` *`lambda`*. The data set it is identified by its sequence number through subcommand `>>>> INDEX`.

Example
```
> PARAMETER
  >> BOX-COX
     >>> MODEL
         >>>> INDEX: 1
         >>>> VALUE
         >>>> RANGE: -1 1
         >>>> VARIATION: 0.1
         <<<<
     <<<
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT                  : A1125
         >>>> Read DATA from FILE : pres.col
         >>>> standard DEVIATION  : 200.0 Pa
         >>>> BOX-COX             : 0.1
         <<<<
```

See Also
```
>>>> BOX-COX
```

Command
>>>> BOX-COX: *lambda*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
none

Description
This command sets parameter λ of the Box-Cox transformation. To address a violation of the homoscedasticity assumption, the Box-Cox transformation [*Box and Cox*, 1964] can be applied to the measured and simulated data as follows:

$$\tilde{z} = \begin{cases} \dfrac{(z+a)^{\lambda} - 1}{\lambda \cdot g^{\lambda - 1}} & \text{if } \lambda \neq 0 \\ g \cdot \ln(z+a) & \text{if } \lambda = 0 \end{cases}$$

Here, g is the geometric mean of the data in the data set, and a is an internally calculated shift parameter, ensuring that the argument to the power function is positive.

Even though the data of the corresponding simulation result will be transformed, the standard deviation has to refer to the untransformed data; it will be transformed internally to calculate the appropriate weight of the residuals. The Box-Cox transformation parameter can be estimated using command >> BOX-COX.

Example

```
> OBSERVATION

  >> PRESSURE
     >>> ELEMENT                  : A1125
         >>>> Read DATA from FILE : pres.col
         >>>> standard DEVIATION  : 200.0 Pa
         >>>> BOX-COX             : 0.1
         <<<<
     <<<
```

See Also
>> BOX-COX, >>>> LOGARITHM (o)

Command
```
>> CAPACITY
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the rock grain specific heat (TOUGH2 variable *SPHT(NMAT)*).

Example
```
> PARAMETER
  >> heat CAPACITY
     >>> ROCK type      : GRANI
         >>>> VALUE
         >>>> RANGE     : 600.0  3000.0 [J/kg/C]
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>> CAPILLARY
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> DEFAULT
>>> MATERIAL
```

Description
This command selects a parameter of the capillary pressure function (TOUGH2 variable *CP(IPAR,NMAT)*) of a certain rock type, or a parameter of the default capillary pressure function (TOUGH2 variable *CPD(IPAR)*). Use command `>>>> INDEX` to select the parameter index *IPAR*. The physical meaning of the parameter depends on the type of capillary pressure function selected in the TOUGH2 input file, variable *ICP* and *ICPD*, respectively. The admissible range should be specified explicitly to comply with parameter restrictions (see *Pruess* [1987], Appendix B).

Example
```
> PARAMETER
  >> parameter of CAPILLARY pressure function
     >>> DEFAULT
         >>>> ANNOTATION    : Slr
         >>>> INDEX     CPD(: 2)
         >>>> VALUE
         >>>> RANGE         : 0.01 0.99
         <<<<
     >>> MATERIAL: SAND1 SAND2
         >>>> ANNOTATION    : vG alpha [Pa^-1]
         >>>> INDEX no.     : 3
         >>>> LOGARITHM
         >>>> RANGE         : -5.0 -1.0
         <<<<
     <<<
  <<
```

See Also
```
>> RELATIVE
```

Command
```
>>> CAUCHY
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects an objective function that corresponds to a Cauchy or Lorentzian distribution, i.e., the probability density function of the residuals reads:

$$\varphi(r_i) \sim \frac{1}{1+\frac{1}{2}\left(\frac{r_i}{\sigma_i}\right)^2}$$

This distribution exhibits more extensive tails compared to the normal distribution, and leads therefore to a more robust estimation if outliers are present. The objective function to be minimized is given by the following equation:

$$S = \sum_{i=1}^{m} \log\left(1+\frac{1}{2}y_i^2\right)$$

with

$$y_i = \frac{r_i}{\sigma_i}$$

This objective function can be minimized using the standard Levenberg-Marquardt algorithm, which is designed for a quadratic objective function. The objective function can be reasonably well approximated by a quadratic function, so that the Levenberg-Marquardt algorithm is usually quite efficient.

Example
```
> COMPUTATION
  >> OPTION
     >>> measurement errors follow CAUCHY distribution
     <<<
  <<
```

See Also
```
>>> ANDREWS, >>> L1-ESTIMATOR, >>> LEAST-SQUARES,
>>> QUADRATIC-LINEAR
```

Command
```
>>> CENTERED
```

Parent Command
```
>> JACOBIAN
```

Subcommand
-

Description
This command calculates elements of the Jacobian matrix by means of centered finite difference quotients:

$$J_{ij} = \frac{\partial z_i}{\partial p_j} \approx \frac{z_i(\mathbf{p}; p_j + \delta p_j) - z_i(\mathbf{p}; p_j - \delta p_j)}{2\delta p_j}$$

The evaluation of the Jacobian thus requires $2n+1$ TOUGH2 simulations, where n is the number of parameters. The size of the perturbation δ can be controlled using command `>>> PERTURB`. Centered finite differences are more accurate than forward finite differences, but computationally twice as expensive (also see command `>>> FORWARD`).

Example
```
> COMPUTATION
  >> JACOBIAN
     >>> use CENTERED finite difference quotient
     >>> PERTURBation factor is: 0.005 times the parameter
     <<<
  <<
```

See Also
```
>>> FORWARD (j), >>> PERTURB
```

Keyword
CHANGE/DELTA

Parent Command
Any second-level command in block > OBSERVATION

Subcommand
-

Description
Subtracts the value calculated at the first calibration time from the values calculated at subsequent calibration times, allowing the use of temporal changes rather than absolute values as calibration data. The first calibration point by definition has a residual of zero and is thus not included in the objective function and statistical analyses. Alternatively, a reference value for the first calibration point can be provided through command >>>> REFERENCE, in which case the difference to this reference value is calculated, and the first calibration point is included in the analysis (see example).

Note that the measured data are expected to reflect relative changes to the initial value (rather than absolute values), which can be accomplished by subtracting the value measured at the first calibration time using command >>>> SHIFT.

This keyword cannot be combined with third-level keywords PROFILE, SECTION, or MAP, i.e., it can only be used with time-series data, not spatial data.

Example
```
> OBSERVATION

     (pumping starts at t=3600))
  >> TIMES: 50 LOGARITHMICALLY SPACED
     3610. 14400.

  >> DELTA(PRESSURE)
     >>> ELEMENT: WEL_1
         >>>> ANNOTATION  : DRAWDOWN
         >>>> REFERENCE P.:  1.4753E5  (dP = P - Pref)
         >>>> SHIFT Pmeas : -1.4753E5  (P measured at t=3600)
         >>>> DATA on FILE: Pmeas.dat
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>> CHARACTERISTIC
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command generates a file in the format specified by command `>>> FORMAT` containing the characteristic curves (relative permeability and capillary pressure functions) of all rock types used in the TOUGH2 model. The file name contains the string "*_ch*".

Example
```
> COMPUTATION
  >> OUTPUT
     >>> generate file with CHARACTERISTIC curves
     >>> in : TECPLOT FORMAT
     <<<
  <<
```

See Also
```
>>> FORMAT
```

Command
>>>> COLUMN: *itime idata* (*istd_dev*)

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command identifies the column holding the time and observed value in the data definition block (see command >>>> DATA). By default, transient observations must be provided in two columns, where the first column contains the observation time, and the second column contains the corresponding measurement. Deviations from this format must be indicated by providing the numbers of the columns, *itime* and *idata*, holding time and data information, respectively. If a third integer *istd_dev* is provided, an additional column is expected holding the standard deviation of the corresponding measurement. This allows one to specify individual standard deviations for each data point (the subcommand >>>> DEVIATION assigns a single standard deviation to all data points of a data set). The columns following command >>>> DATA are read in free format. If a column contains non-numeric characters, command >>>> FORMAT must be used.

Example
```
> OBSERVATION
  >> CAPILLARY PRESSURE
     >>> ELEMENT                : A1__1
         >>>> COLUMN            : 2 3
         >>>> skip              : 3 HEADER lines
         >>>> conversion FACTOR : -100.0  [hPa] - [-Pa]
         >>>> DATA               (time is in MINUTES)
-------------------------------------------------------------
  # Time [min]  Cap. Pres, [hPa] FlowmeterX Flow Rate [mg/s]
-------------------------------------------------------------
  1       5.0   0.1698331055E+02 FlowmeterA 0.9869399946E+01
  2      10.0   0.2075763428E+02 FlowmeterA 0.1039689596E+02
  3      15.0   0.2357142822E+02 FlowmeterA 0.1162893932E+02
  4      20.0   0.2529052490E+02 FlowmeterA 0.1353439620E+02
  .      ....   ................ .......... ................
         >>>> standard DEVIATION: 0.5 [hPa]
         <<<<
     <<<
```

See Also
>>>> DATA, >>>> DEVIATION (o), >>>> FORMAT, >>>> HEADER, >>>> PICK

Command

```
>>>> COMPONENT comp_name/: icomp
```

Parent Command

```
>>> ELEMENT
>>> SOURCE (o)
```

Subcommand

-

Description

This command identifies a component either by its name (*comp_name*) or the component number (*icomp*). A list of allowable component names for the given EOS module can be obtained from the header of the iTOUGH2 output file.

Example

```
> OBSERVATION
  >> CONCENTRATION
     >>> ELEMENT: ZZZ99
         >>>> ANNOTATION: TCE concentration
         >>>> COMPONENT No.: 3  (=VOC)
         >>>> dissolved in LIQUID PHASE
         >>>> DATA on FILE: tce.dat
         >>>> standard DEVIATION: 1.0E-6
         <<<<
     <<<
  <<
```

See Also

```
>>>> PHASE
```

Command
```
>> COMPRESSIBILITY
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the pore space compressibility c_ϕ (TOUGH2 variable *COM(NMAT)*) of a certain rock type. Under fully liquid-saturated conditions, pore space compressibility estimates c_ϕ can be converted to a specific storage coefficient S_s [m^{-1}] as follows:

$$S_s = \phi \cdot \rho \cdot g\left(c_l + c_\phi\right)$$

where ϕ is porosity, ρ is density of water, g is gravitational acceleration, and $c_l \approx 4.4 \cdot 10^{-10}$ [Pa^{-1}] is water compressibility. Similarly, estimation of c_ϕ for grid blocks representing a well or borehole is useful for the determination of a dimensionless wellbore storage coefficient C_{bh}:

$$C_{bh} = \phi_{bh} \cdot \rho \cdot g \cdot V_{bh}\left(S_l \cdot c_l + S_g \cdot c_g + c_\phi\right)$$

where ϕ_{bh} and V_{bh} are the porosity and volume of the grid block representing the well, S_l and S_g are the liquid and gas saturation, and c_g is gas compressibility which is approximately $1/p$.

Example
```
> PARAMETER
  >> pore space COMPRESSIBILITY
     >>> MATERIAL: BOREH
         >>>> ANNOTATION: Wellbore storage
         >>>> LOGARITHM
         >>>> RANGE     : -10.00  -7.0
         <<<<
     >>> MATERIAL: SKIN_ ROCK_ BOUND
         >>>> ANNOTATION: Storativity
         >>>> LOGARITHM
     <<<
  <<
```

See Also
-

Command
```
> COMPUTATION
```

Parent Command
-

Subcommand
```
>> CONVERGE
>> ERROR
>> JACOBIAN
>> OPTION
>> OUTPUT
```

Description
This is the first-level command for specifying a number of computational parameters, convergence criteria, program options, and output formats. The general format is as follows:

```
> COMPUTATION
  >> specify various program OPTIONS
  >> specify CONVERGEnce criteria
  >> specify parameters for calculating JACOBIAN matrix
  >> specify parameters for ERROR analysis
  >> specify OUTPUT formats
  <<
```

Example
```
> COMPUTATION
  >> CONVERGENCE criteria
     >>> perform : 5 ITERATIONS
     <<<
  >> JACOBIAN
     >>> use CENTERED finite difference quotient with
     >>> a relative parameter PERTURBation of : 0.5 %
     <<<
  >> program OPTIONS
     >>> use LEAST-SQUARES objective function (default)
     >>> allow the simulation to reach STEADY state
     <<<
  >> OUTPUT
     >>> generate PLOTFILE for: TECPLOT
     >>> print all times in HOURS
  <<
```

See Also
-

Command

```
>> CONCENTRATION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> ELEMENT
```

Description

This command selects as an observation type the concentration [kg/m^3] of a component in a given phase. Concentration is defined as the product of mass fraction of component *icomp* in phase *iphase* times density of phase *iphase*. This observation type refers to an element. Component number *icomp* or component name *comp_name*, and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the header of the iTOUGH2 output file, and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively.

Example

```
> OBSERVATION
  >> CONCENTRATION of BRINE in LIQUID
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> CONCENTRATION of COMPONENT No.: 2 in PHASE No.: 2
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> CONCENTRATION
     >>> ELEMENT: A1__1
         >>>> COMPONENT: 2
         >>>> LIQUID PHASE
```

See Also

`>> MASS FRACTION`, `>> MOLE FRACTION`

Command
```
>> CONDUCTIVITY (WET/DRY)
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the formation heat conductivity under fully liquid-saturated (keyword `WET`, default, TOUGH2 variable *`CWET(NMAT)`*) or desaturated conditions (keyword `DRY`, TOUGH2 variable *`CDRY(NMAT)`*).

Example
```
> PARAMETER
  >> heat CONDUCTIVITY under DRY conditions
     >>> ROCK type      : GRANI
         >>>> VALUE
         >>>> RANGE     : 0.5 5.0 [W/m/C]
         <<<<
     <<<
  <<
```

See Also
-

Command

```
>>> CONNECTION: elem1 elem2 (elem_i elem_j ...)
                (++/+-/++ iplus)
>>> CONNECTION COORDINATES: X Y Z
>>> CONNECTION COORDINATE BOX : Xmin Ymin Zmin Xmax Ymax Zmax
>>> CONNECTION COORDINATE ELLIPSOID : Xc Yc Zc DX DY DZ
>>> CONNECTION COORDINATE CYLINDER : Xs Ys Zs Xe Ye Ze Radius
>>> CONNECTION COORDINATE CUBE: Xc Yc Zc DX DY DZ
>>> CONNECTION PROFILE/CROSS-SECTION/MAP
```

also:

```
>>> MATERIAL: material1 material2 (material3 material4 ... ...)
```

Parent Command

```
>> FLOW
```

Subcommand

any fourth-level command in block `> OBSERVATION`

Description

This command reads pairs of element names defining a connection. By default, element names are designated by a three-character/two-integer code name. Blanks in the element names as printed in the TOUGH2 output file must be replaced by underscores (e.g., an element name specified in the TOUGH2 input file as '`B 007`' is printed as '`B 0 7`' to the TOUGH2 output file. Therefore, it has to be addressed in the iTOUGH2 input file as '`B_0_7`'). Multiple connections can be specified, and iTOUGH2 calculates the sum or mean of all flow rates (see subcommands `>>>> SUM` and `>>>> AVERAGE`, respectively).
A sequence of *iplus* connections can be generated where the number of the first and/or the second element is increased by 1. If only the first (second) element in a sequence of connections should be increased, use +− (−+). If both elements are to be increased, use + (or ++). The following two command lines are thus identical:

```
>>> CONNECTION: AA__1 BB_15   -+ 2
>>> CONNECTION: AA__1 BB_15   AA__1 BB_16   AA__1 BB_17
```

If keyword `COORDINATES` is present, coordinates can be specified, and the connection closest to these coordinates will be identified.

If keywords `COORDINATE BOX`, `COORDINATE ELLISPODE`, `COORDINATE CYLINDER`, or `COORDINATE CUBE` are present, all connections within the specified region will be included in the element list. The region is defined according to the following table. If keyword `COMPLEMENT` or `OUTSIDE` is present, the complementary region is

selected (i.e., all connections outside the defined geometry). Element coordinates must be present in columns 51–80 of the TOUGH2 ELEME block.

KEYWORD	*XREGION(i)*								
	1	2	3	4	5	6	7	8	9
(none)	X	Y	Z	-	-	-	-	-	-
BOX	X_{min}	Y_{min}	Z_{min}	X_{max}	Y_{max}	Z_{max}	aximuth	dip	plunge
ELLIPSOID	X_{center}	Y_{center}	Z_{center}	$X_{semi-axis}$	$Y_{semi-axis}$	$Z_{semi-axis}$	aximuth	dip	plunge
CYLINDER	X_{start}	Y_{start}	Z_{start}	X_{end}	Y_{end}	Z_{end}	radius	-	-
CUBE	X_{center}	Y_{center}	Z_{center}	$X_{half-length}$	$Z_{half-length}$	$Z_{half-length}$	aximuth	dip	plunge

Add keyword ANGLE or ROTATE if the region is not aligned with the coordinate axes, and provide azimuth, dip, and plunge correction angles.

Keywords CONSTANT (default), LINEAR, SPHERICAL, EXPONENTIAL, or RAMP may be added to weigh the contribution of each observation according to its distance *d* from the center of the region.

$$\omega = 1 \qquad \text{(constant)}$$

$$\omega(d) = 1 - d \qquad \text{(linear)}$$

$$\omega(d) = 1 - \left(\frac{3}{2} d - \frac{1}{2} d^3 \right) \qquad \text{(spherical)}$$

$$\omega(d) = a^{-d} \qquad \text{(exponential)}$$

$$\omega(d) = \min(1, (1 - d)/a) \qquad \text{(constant-linear ramp)}$$

If keyword PROFILE, CROSS-SECTION, or MAP is present, the data (which must be provided in an external file, see command >>>> DATA FILE: *filename*) are organized spatially by lists of coordinates followed by columns of measured values for each survey time. The data will be assigned to the nearest connection (element coordinates must be present in columns 51–80 of the TOUGH2 ELEME block.). Keyword PROFILE supports one-dimensional, vertical data profiles (i.e., Z-coordinates), whereas CROSS-SECTION and MAP both require X-, Y-, and Z-coordinates. Profile data are given as follows:

```
NZ  NT
XCoord  YCoord
      Time(1)    Time(2)    ...  Time(NT)
Z(1)  Data(1,1)  Data(1,2)  ...  Data(1,NT)
Z(2)  Data(2,1)  Data(2,2)  ...  Data(2,NT)
...   ...        ...        ...  ...
Z(NZ) Data(NZ,1) Data(NZ,2) ...  Data(NZ,NT)
```

Two- or three-dimensional data are given as follows:

```
NP  NT
                    Time(1)    Time(2)    ...  Time(NT)
X(1)  Y(1)  Z(1)    Data(1,1)  Data(1,2)  ...  Data(1,NT)
X(2)  Y(2)  Z(2)    Data(2,1)  Data(2,2)  ...  Data(2,NT)
...   ...   ...     ...        ...        ...  ...
X(NP) Y(NP) Z(NP)   Data(NP,1) Data(NP,2) ...  Data(NP,NT)
```

If command `>>> MATERIAL` is used, all connections across the interface between pairs of materials are selected. The sign of the flow rate is automatically adjusted so that flow across the interface is defined using a consistent definition of direction, regardless of the order of elements in the connection.

Example

```
> OBSERVATION
  >> LIQUID FLOW RATE
     >>> list of CONNECTIONS: ELM_1 ELM_2 + 48
         >>>> ANNOTATION    : Boundary flux
         >>>> take SUM of flow rates across 49 connections
         >>>> DATA on FILE  : flow.dat
         >>>> RELATIVE error: 10 %
         <<<<
     <<<

  >> GAS FLOW RATE
     >>> all CONNECTIONS in ROTATED COORDINATE BOX:  100  300  -1  &
                                                     250  700   1  &
                                                      60    0   0
         >>>> ANNOTATION    : Surface gas release
         >>>> take SUM of flow rates across surface
         >>>> HEADER  : 2
         >>>> COLUMNS : 1 3
         >>>> DATA on FILE  : flow.dat
         >>>> RELATIVE error: 10 %  + 0.01
     <<<

  >> RN1 FLOW RATE across...
     >>> MATERIAL boundary between: WASTE BENTO
```

See Also

`>>>> ABSOLUTE, >>>> AVERAGE, >>>> SUM`

Command
>>> CONSECUTIVE: *max_iter1*

Parent Command
>> CONVERGE

Subcommand
-

Description
By default, TOUGH2 simulations are stopped if 10 consecutive time steps converge with a single Newton-Raphson iteration because no update of primary variables occurs. This command allows changing the maximum number of allowable time steps with ITER=1 to *max_iter1*.
Consecutive time steps with no update of primary variable occur if:

(1) steady state is reached;
(2) calibration or printout times are too narrowly spaced;
(3) the maximum time step size (TOUGH2 variable *DELTMX*) is too small;
(4) the initial time step (TOUGH2 variable *DELTEN* or *DLT(1)*) is too small;
(5) a small time step is taken to land on a calibration or printout time.

Only (1) is an acceptable TOUGH2 convergence (see command >>> STEADY-STATE). All the other reasons may lead to premature termination of a TOUGH2 simulation. Convergence problems are more often encountered in iTOUGH2 than in a standard TOUGH2 simulation because many parameter combinations are submitted. This command makes TOUGH2 more tolerant of this kind of convergence failure. It is important, however, that *max_iter1* is only increased to overcome temporary convergence problems during the optimization, i.e., the final parameter set should yield a TOUGH2 simulation without convergence problems. Calibration points should not be spaced too narrowly in time (see command >> TIME). Command >>> ADJUST can be used to overcome problem (4). Note that a special time stepping procedure is incorporated in iTOUGH2 to avoid problem (5).

Example
```
> COMPUTATION
  >> CONVERGE
     >>> accept: 20 CONSECUTIVE time steps
         that converge on ITER=1
     <<<
  <<
```

See Also
>> TIME, >>> ADJUST, >>> REDUCTION

Command
>> CONTENT (*phase_name*/PHASE: *iphase*)

Parent Command
> OBSERVATION

Subcommand
>>> ELEMENT

Description
This command selects as a parameter the content of phase *iphase* as an observation type. Phase content is defined as the product of saturation and porosity.
The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the iTOUGH2 header, and can be specified either on the command line or using subcommand >>>> PHASE.
Estimating phase content and phase saturation is identical only if porosity remains constant. Porosity can be variable (i) due to compression of the pore space (i.e., if TOUGH2 variable *COM(NMAT)* is not zero), and (ii) if porosity is one of the parameters to be estimated and is updated during the inversion.

Example
```
> OBSERVATION
  >> TIME: 1 point at steady state
     1.0E20
  >> LIQUID CONTENT or
     CONTENT in PHASE No.: 2
     >>> ELEMENT: A1__1
         >>>> ANNOTATION: Water content
         >>>> FACTOR    : 0.01  (data given in %)
         >>>> one steady-state DATA point
              0.0    23.0
              1.0E50 23.0
         >>>> DEVIATION:  5.0 %
         <<<<
     >>> ELEMENT: A1__1
         >>>> ANNOTATION: Gas content
         >>>> GAS PHASE (overwrites phase
                         specified on command line)
         >>>> NO DATA, just for plotting
         <<<<
     <<<
  <<
```

See Also
-

Command

```
>> CONVERGE
```

or

```
>> STOP
```

or

```
>> TOLERANCE
```

Parent Command

```
> COMPUTATION
```

Subcommand

```
>>> ADJUST
>>> CONSECUTIVE
>>> FORWARD
>>> INCOMPLETE
>>> INPUT
>>> ITERATION
>>> LEVENBERG
>>> LIST
>>> MARQUARDT
>>> REDUCTION
>>> SIMULATION
>>> STEP
>>> UPHILL
>>> WARNING
```

Description

This is the parent command of a number of subcommands that deal with tolerance measures and convergence criteria for the inversion and, to a certain extent, the TOUGH2 simulation.

Example

```
> COMPUTATION
  >> CONVERGEnce criteria
     >>> ignore WARNING messages, then
     >>> perform : 5 ITERATIONS
     >>> stop if more than : 5 unsuccessful UPHILL steps
     >>> and : 20 time step REDUCTIONS
     >>> accept : 6 INCOMPLETE TOUGH2 runs
     >>> the initial LEVENBERG parameter shall be : 1.0
     >>> MARQUARDT parameter is: 10.0
     <<<
```

See Also

```
>> OPTION
```

Command
>>>> CORRELATION: (-)*rcorr*

Parent Command
>>> SELECT

Subcommand
-

Description
This command defines one of the criteria used for automatic parameter selection. It examines the ratio between the apparent conditional standard deviation σ_p * and the joint standard deviation σ_p as a measure of overall parameter correlation (since the calculation is performed far from the minimum, the standard deviations cannot be interpreted as actual estimation uncertainties):

$$\chi = \frac{\sigma_p{}^*}{\sigma_p} \quad 0 < \chi \le 1$$

Those parameters with a ratio larger than |*rcorr*|, i.e., the most independent parameters, are selected. Strongly correlated parameters are (temporarily) excluded from the optimization process.
If a negative value is given for *rcorr*, the selection criterion is relaxed with each iteration *k*, and reaches zero for the last iteration *max_iter*, i.e., all parameters are selected for the final step.

$$rcorr_k = |rcorr| \cdot \left(1 - \frac{k}{\max\_iter}\right)$$

The choice for *rcorr* depends on the number of parameters *n* specified in block > PARAMETER. The more parameters are estimated simultaneously, the higher are parameter interdependencies. This fact should be acknowledged by specifying a smaller value for |*rcorr*| if *n* increases.

Example
```
> COMPUTATION
  >> OPTION
     >>> SELECT parameter automatically every
         >>>> : 3 ITERATIONS
         >>>> use CORRELATION criterion with rcorr : -0.10
         <<<<
      <<<
  <<
```

See Also
>>>> ITERATION (s), >>>> SENSITIVITY

Command
```
>> COVARIANCE (FILE: file_name)
```

Parent Command
```
> OBSERVATION
```

Subcommand
-

Description
This command reads the diagonal elements of the *a priori* covariance matrix $\mathbf{C}_{zz}$. Usually the variances of the observations are specified separately for each data set using command `>>>> DEVIATION (o)`, or they are provided as a separate column along with the data (see commands `>>>> COLUMN` and `>>>> DATA`). As an alternative, one can assign or overwrite variances using the second-level command `>> COVARIANCE`, followed by two columns holding index i and variance c_{ii}. Since this command addresses elements of the assembled covariance matrix, the user must provide the index that corresponds to the position of the observation in vector $\mathbf{z}$. This information is best retrieved from the iTOUGH2 output file after running one forward simulation. Note that the index changes whenever the number of observations, parameters, or calibration times is changed. The variances can also be read from a covariance file which has to contain three columns, holding index i, index j, and the (co-)variance c_{ij} (this is the same format as the one on the covariance file generated by command `>>> COVARIANCE`). Despite the fact that two indexes must be provided, only diagonal elements, i.e., $i = j$, will be accepted as input.

Example
```
> OBSERVATION
  >> : 30 EQUALLY space calibration TIMES in MINUTES between
     3.0 90.0
  >> plus : 1 steady-state TIME near
     86400.0 seconds
  >> PRESSURE
     >>> ELEMENT: A1__1
         >>>> all DATA on FILE: pres.dat
         >>>> standard DEVIATION:  2000.0 Pa
         <<<<
     <<<
  >> COVARIANCE
     32  1.0E4
     (changes one element of the covariance matrix to
     increase the weight of the steady-state data point)
  <<
```

See Also
```
>>> COVARIANCE, >>>> COLUMN, >>>> DEVIATION, >>>> DATA
```

Command
```
>>> COVARIANCE
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command generates a file with extension "*.cov* " with the covariance matrix of the calculated system response:

$$\mathbf{C}_{\hat{z}\hat{z}} = \mathbf{J}\mathbf{C}_{pp}\mathbf{J}^T = s_0^2 \cdot \mathbf{J}\left(\mathbf{J}^T\mathbf{C}_{zz}^{-1}\mathbf{J}\right)^{-1}\mathbf{J}^T$$

Note that $\mathbf{C}_{\hat{z}\hat{z}}$ is a square matrix of dimension $m \times m$.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print COVARIANCE matrix of calculated
         system response
     <<<
  <<
```

See Also
-

Command

```
>> CUMULATIVE (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> SOURCE
```

Description

This command selects as an observation type the cumulative injection or production of component *icomp* or phase *iphase*. This observation type refers to a sink or source code name. It can be used when time-dependent generation rates are to be estimated where the total amount of injected or produced fluid is approximately known, or if the total generation rate is prescribed in block GENER, but the phase composition of the produced fluid is variable and sensitive to the parameters of interest. Finally, the cumulative amount of injected or produced fluid can be used as an observable variable for wells on deliverability.

Note that the cumulative mass of a phase produced in an element strongly depends on the composition of the produced fluid mixture according to the options provided by TOUGH2 flag *MOP(9)*.

Component number *icomp* or component name *comp_name,* and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the iTOUGH2 header and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively. If neither a phase nor component number is specified, the total, cumulative mass of all phases or components will be calculated.

Example

```
> OBSERVATION
  >> CUMULATIVE METHANE produced
     >>> SOURCE: RW__1
         >>>> ANNOTATION    :  Total methane [l]
         >>>> FACTOR        : -7.67358E-04   [l] - [kg]
         >>>> DATA from FILE: tot_ch4.dat    [HOUR]
         >>>> DEVIATION     :   5.0          [l]
         <<<<
     <<<
  <<
<
```

See Also

-

Command
>>>> DATA (NO DATA/ZERO DATA/FILE: *file_name*) (*time_unit*)

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command reads a list of observation times and the corresponding data. Each list will be referred to as a data set in the iTOUGH2 output file. An annotation is generated for each data set, or a string can be supplied by the user for its identification (see command >>>> ANNOTATION).
Data can be supplied either directly following the command line or on a separate file (use keyword FILE followed by the name of the data file after a colon).
Time and data must be arranged in columns. They are read in free format. Data are accepted until a FORTRAN input error occurs. The total number of data points read by iTOUGH2 is printed to the iTOUGH2 output file and should be checked for consistency.
By default, the first column is expected to hold the observation times, and the second column the data values. Deviations from this format are possible (see commands >>>> COLUMN, >>>> FORMAT, >>>> SET (o), and >>>> HEADER for details). Data can also be represented by a polynomial or a user-specified function (see commands >>>> POLYNOM and >>>> USER, respectively).
Observation times do not need to coincide with the calibration times defined by command >> TIME (o). Linear interpolation is performed for calibration times that fall between observation times. This requires, however, that the first observation time is earlier than the first calibration time, and that the last observation time is later than the last calibration time. If this condition is not met, command >>>> WINDOW should be used. Only one time window can be specified for each data set, i.e., multiple data sets must be provided if multiple time windows are needed.
If time is not given in seconds, the appropriate time unit (MINUTE, HOUR, DAY, WEEK, MONTH, YEAR) must be specified on the command line. If the units of the data points are different from the standard units used in TOUGH2, a conversion factor must be provided through command >>>> FACTOR.
If no observed data are available (e.g., when performing sensitivity or data-worth analyses, or when using iTOUGH2 for generating time series plots), the (optional) command >>>> NO DATA can be used. In this case, even though the computed results are shown in the residual analysis, the residuals do not contribute to the objective function.
If all actual or virtual data values are zero, command >>>> ZERO DATA can be used.

Example

```
> OBSERVATION
  >> GAS PRESSURE
     >>> in ELEMENT: A1__1
         >>>> DATA in default format, time (s) vs. pres. (Pa)
                 1.0   100000.0
                10.0   101343.8
                20.0   105991.3
                30.0   108965.9
                60.0   115003.8
                ....   ........
              3600.0   218762.0
         >>>> standard DEVIATION: 5000.0 Pa
         <<<<
     <<<

  >> LIQUID FLOW rate
     >>> CONNECTION B1__1 B1__2
         >>>> HEADER contains : 3 lines to be skipped
         >>>> time and value are in COLUMNS: 3  6
         >>>> conversion FACTOR: -1.6667E-5
              (converts from [ml/min] to [-kg/s])
         >>>> use SET No. : 3
         >>>> DATA on FILE : flow.dat with time in MINUTES
         >>>> RELATIVE measurement error: 5.0 % + 0.1
         <<<<
     <<<

  >> BRINE CONCENTRATION in LIQUID PHASE
     >>> ELEMENT: C1__1
         >>>> POTENTIAL observation
         >>>> NO DATA, this is for observation-worth analysis
         <<<<
     <<<
  <<
```

See Also

```
>>>> ANNOTATION, >>>> COLUMN, >>>> FACTOR, >>>> FORMAT,
>>>> HEADER, >>>> PICK, >>>> POLYNOM, >>>> POTENTIAL,
>>>> SET (o), >>>> USER, >>>> WINDOW, >>> WORTH (o, op)
```

Command

```
>>>> DECPOINT: POINT/NOPOINT
```

Parent Command

```
>>> PEST
```

Subcommand

-

Description

iTOUGH2 capabilities can be applied to non-TOUGH2 models using the PEST interface [*Doherty*, 2002]. By selecting the keyword `NOPOINT`, the decimal point in the representation of a parameter in the input file is omitted, potentially increasing the accuracy of a parameter value. However, this should be done with great caution, as fields read by FORTRAN programs with format specifiers such as "(F6.2)" or "(E8.2)" will insert their own decimal points; for details, see Section 3.2.6 of *Doherty* [2002]. Therefore, `POINT` is the default option.

Example

```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE  :     pointnopoint.exe
         >>>> TEMPLATE    : 1
              pointnopoint.tpl  whatsthepoint.in

         >>>> INSTRUCTION : 1
              pointnopoint.ins  pointless.out

         >>>> DECPOINT: NOPOINT
         <<<<
```

See Also

-

Command
```
>>> DEFAULT
```
or
```
>>> MATERIAL: DEFAU
```

Parent Command
```
>> CAPILLARY
>> INITIAL
>> RELATIVE
```

Subcommand
any fourth-level command in block `> PARAMETER`

Description
Parameters of the default relative permeability and capillary pressure functions (TOUGH2 block RPCAP) or default initial conditions (TOUGH2 block PARAM.4) are addressed by command `>>>> DEFAULT`. Alternatively, a material name `DEFAU` can be provided following command `>>>> MATERIAL.`

Example
```
> PARAMETER
  >> INITIAL PRESSURE
     >>> DEFAULT initial pressure (block PARAM.4)
         >>>> ANNOTATION: Init. Formation Pres.
         >>>> GUESS     : 1.5E5
         <<<<
     <<<

  >> RELATIVE PERMEABILITY FUNCTION
     >>> MATERIAL: BOREH
         >>>> ANNOTATION : Sgr borehole
         >>>> PARAMETER #: 2
         <<<<
     >>> MATERIAL: BOUND DEFAU
         >>>> ANNOTATION : Sgr elsewhere
         >>>> PARAMETER #: 2
         <<<<
     <<<
  <<
```

See Also
```
>>> MATERIAL, >>> MODEL
```

Command
`>>> DELTFACT:` *`deltfact`*

Parent Command
`>> CONVERGE`

Subcommand
-

Description
In TOUGH2, time step size can be controlled in various ways (see *Pruess* [1987] for a description of input variables *`DELTEN`*, *`DLT`*, *`DELTMX`*, *`NOITE`*, *`REDLT`*, and *`MOP(16)`*). The time step is also automatically adjusted in order for the simulation time to land on any of the specified calibration or printout times. This may lead to very small time steps or even convergence failures (see command `>>> CONSECUTIVE`).
In iTOUGH2, the proposed time step is increased up to a factor of *`deltfact`* (default: 1.1) if this increase allows the simulation to reach the next calibration or printout time. This may lead to time stepping different from standard TOUGH2.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> DELTFACT: 1.0  (as in standard TOUGH2)
     >>> allow : 20 CONSECUTIVE time steps
         converging in 1 iteration
     <<<
   <<
```

See Also
`>>> CONSECUTIVE`

Command
`>>> DESIGN`

Parent Command
`>> OPTION`

Subcommand
-

Description
(synonym for command `>>> SENSITIVITY`)

Example
(see command `>>> SENSITIVITY`)

See Also
`>>> SENSITIVITY`

Command
```
>>> DETREND
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints detrending information.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print DETRENDING infomration.
     <<<
  <<
```

See Also
```
>>>> DETREND
```

Command
>>>> DETREND: *-timewindow* (*time_units*) / *-npoints*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command detrends observed and calculated system responses by subtracting a moving average of the values either within a given time window (given as a positive real *timewindow*) or within a given number of points (given as a negative integer *npoints*).

Example

```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT                   : A1125
         >>>> Read DATA from FILE  : pres.col
         >>>> standard DEVIATION   : 200.0 Pa
         >>>> DETREND              : 24.0 HOURS
         <<<<
```

See Also
>>>> REGRESSION, >> DRIFT

Command
`>>>> DEVIATION: sigma`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command specifies the standard deviation σ_{p_i} of the initial parameter guess. Prior information about model parameter *i* will be weighted by $1/\sigma_{p_i}$, i.e., the difference between the prior information value p_i* and the estimate $\hat{p}_i$ contributes to the objective function as follows:

$$\text{weighted squared residual from prior information} = \frac{(p_i^* - \hat{p}_i)^2}{\sigma_{p_i}^2}$$

Commands for specifying the standard deviation are:

`>>>> DEVIATION:` σ_{p_i}
`>>>> VARIANCE:` $\sigma_{p_i}^2$
`>>>> WEIGHT:` $1/\sigma_{p_i}$

By default, prior information is not weighted, i.e. $\sigma_{p_i}^2 = \infty$.
The standard deviation reflects the uncertainty associated with the initial guess. If the initial guess is to be weighted, prior information should originate from an independent source. For example, if porosity will be estimated based on transient pressure data, the prior information value should be taken from a "direct" porosity measurement, e.g. using mercury-porosimetry or oven-drying methods. In these cases, the measured parameter values p_i* are considered to be additional data points which serve as a physical plausibility criterion for the estimate $\hat{p}_i$. The p_i* values, along with the observations of the system state z_i*, are then weighted according to their uncertainties (see `>>>> DEVIATION (o)`). Note that the relative weighting between prior information and the observations z_i* depends on the number of calibration points selected. If many transient data points are available, a smaller standard deviation σ_{p_i} may be specified to increase the relative weight of prior information.
In many cases, appropriately weighting the initial guess makes an ill-posed inverse problem unique. Furthermore, the solution becomes more stable if a parameter is not very sensitive. However, using $1/\sigma_{p_i}$ as a regularization parameter to improve the ability to obtain a unique solution with a poorly conceptualized inverse problem is not recommended. Erratic behavior of a parameter during the inversion should be taken as an indication that the data do not contain sufficient information for the determination of the parameter. Differences between parameter values that are independently determined from laboratory experiments and inverse modeling suggest the presence of a systematic error or scaling problem. These inconsistencies should be resolved rather than averaged out.

The standard deviation σ_{p_i} is also used to scale the columns of the Jacobian matrix. While the solution of the inverse problem is not affected by the choice of the scaling factor, all the qualitative sensitivity measures are directly proportional to σ_{p_i}. If prior information is not weighted, the scaling factor is taken to be 10 % of the respective parameter value. Command `>>>> VARIATION` should be used to change the default scaling factor without concurrently assigning a weight to prior information.
When performing uncertainty propagation analyses, σ_{p_i} designates the parameter uncertainty affecting the model prediction. It is used to generate a set of random parameter values for Monte Carlo simulations, and it represents the standard deviation of a normal distribution if performing linear uncertainty propagation analysis (for more details see commands `>>> MONTE CARLO` and `>>> FOSM`, respectively).

Example

```
> PARAMETER
  >> POROSITY
     >>> MATERIAL: TUFFn
         >>>> PRIOR information : 0.38  (laboratory data)
         >>>> standard DEVIATION: 0.04  (measurement error)
         <<<<
     >>> MATERIAL: ALLUV
         >>>> PRIOR information : 0.30  (from experience)
         >>>> VARIANCE          : 0.01  (of guess)
         <<<<
     >>> MATERIAL: FAULT
         >>>> initial GUESS     : 0.25  (no measurements)
         >>>> WEIGHT            : 0.00  (default)
         >>>> VARIATION         : 0.10  (for scaling of
                                          sensitivity matrix)
         <<<<
     <<<
  <<
```

See Also

`>> GUESS, >>> FOSM, >>> MONTE CARLO, >>>> DEVIATION (o), >>>> PRIOR, >>>> VARIANCE, >>>> VARIATION, >>>> WEIGHT`

Command
`>>>> DEVIATION: sigma (ADD NOISE)`

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
This command specifies the standard deviation σ_z of the observations. The squares of the standard deviations constitute the diagonal elements of the *a priori* covariance matrix.
The specified value *sigma* is assigned to all data points of the corresponding data set. It must be given in the same units as the data (see command `>>>> FACTOR`). Individual values for each calibration point can be explicitly specified using command `>>>> COLUMN` or `>> COVARIANCE`, or are calculated as a fraction of the measured value if using command `>>>> RELATIVE`.
The standard deviation should represent the expected variability of the final residuals. In the absence of modeling errors, the standard deviation is equivalent to the measurement error. A reasonable value can be derived by visual examination of the data, i.e., by estimating the standard deviation of the differences between the observed values and a line representing the expected match; note that this procedure is based on the assumption that time averages can be used to calculate the ensemble average, i.e., that the data set is a result of an ergodic process.
The inverse of the *a priori* covariance matrix is used to weight the fitting error. It also scales observations of different types and units. In the framework of maximum likelihood theory, the covariance matrix constitutes the stochastic model along with the assumption of normality and independence.
The parameter estimates are not affected by the absolute values of σ_z, but only by the ratios $\sigma_{z_i} / \sigma_{z_j}$. It is suggested, however, to use reasonable values that are related to the measurement error. If the final residuals are - on average - significantly larger than the *a priori* specified standard deviations, the Fisher model test fails. The *a posteriori* standard deviations of the final residuals are printed in the output for each data set for comparison purposes.

If keyword `ADD NOISE` is present, Gaussian noise with zero mean and the given standard deviation is added to the data points. This option may be useful if error-free data were synthetically generated.

Alternative commands are:

`>>>> DEVIATION:` σ_{z_i} `(ADD NOISE)`
`>>>> VARIANCE:` $\sigma_{z_i}^2$ `(ADD NOISE)`
`>>>> WEIGHT:` $1/\sigma_{z_i}$ `(ADD NOISE)`

The following command lines are thus equivalent:

```
>>>> standard DEVIATION: 0.1
>>>> VARIANCE: 0.01
>>>> WEIGHT: 10.0
```

Example

```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT : AA412
         >>>> conversion FACTOR: 1E5  from [bar] to [Pa]
         >>>> DATA on FILE: pressure.dat
         >>>> standard DEVIATION: 0.05 [bar]
         <<<<
     <<<
  <<
```

See Also

>> COVARIANCE, >>>> AUTO, >>>> COLUMN, >>>> DEVIATION (p), >>>> RELATIVE, >>>> VARIANCE, >>>> WEIGHT

Command
`>>> DIRECT`

Parent Command
`>> OPTION`
or
`>>> CONVERGE`

Subcommand
-

Description
(synonym for command `>>> FORWARD (o)`)

Example
(see command `>>> FORWARD (o)`)

See Also
`>>> FORWARD (o)`

Command
```
>> DIFFUSION
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects as a parameter the diffusion coefficient (TOUGH2 variable *FDDIAG(IPH,IK)*). The component and phase are identified by two indices in subcommand `>>>> INDEX.`

Example
```
> PARAMETER
  >> DIFFUSION coefficient
     >>> diffusion coefficients refer to the entire MODEL
         >>>> estimate LOGARITHM
         >>>> INDEX: 2 3  (i.e., diffusion coefficient for CO2
                           (IK=3) in the aqueous phase (IPH=2))
         <<<<
     <<<
  <<
```

See Also
-

Command
>> DRAWDOWN (*phase_name*/PHASE: *iphase*)

Parent Command
> OBSERVATION

Subcommand
>>> ELEMENT

Description
This command selects as an observation type the pressure drawdown during a pumping test. This observation type refers to one or more elements. The drawdown is calculated from a reference pressure, which is the pressure at the specified element at the time of the first active calibration point for that data set, i.e., a calibration time greater than the starting time of the simulation must be provided, indicating the beginning of the pumping period; the corresponding observation is a drawdown of zero. Note that the sensitivity of this calibration point is zero by definition, i.e., the first data point is not used for calibration. If drawdown is measured in meters, command >>>> FACTOR must be used to convert the units to Pascals. The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the header of the iTOUGH2 output file. They can be specified either on the command line or using subcommand >>>> PHASE. If no phase is specified, iTOUGH2 takes the pressure drawdown of the first phase, which is usually the reference pressure.

Example
```
> OBSERVATION
  >> TIMES: 1 in [MINUTES]
     10.0  = beginning of pumping
  >> TIMES: 20 LOGARITHMICALLY spaced in [MINUTES]
     11.0  120.0
  >> DRAWDOWN
     >>> ELEMENT: AA__1
         >>>> ANNOTATION    : Pressure drawdown [m]
         >>>> FACTOR        : 9810.  [m] - [Pa]
         >>>> LOGARITHM
         >>>> DATA [MINUTES]
               10.0    0.0   beginning of pumping
               11.0   -0.1
               15.0   -0.3
               ....   .....
         >>>> VARIANCE      :   0.01 [m^2]
         <<<<
```

See Also
>> PRESSURE

Command
```
>> DRIFT
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
>>> SET
```

Description
This command selects as a parameter the slope of a time-dependent trend. The trend is added to the TOUGH2 output referring to a specific data set:

$$z = z_{TOUGH2} + drift \cdot time$$

where (*drift·time*) is the trend added to the calculated TOUGH2 output z_{TOUGH2}. The result z is compared to the measurement z^* of the corresponding data set.
This option allows removal of a trend in the data (for example, a flowmeter may exhibit an unknown offset and time-dependent trend that needs to be estimated). A non-zero value must be provided as initial guess through the iTOUGH2 input file using command `>>>> GUESS`. The data set is identified by number using command `>>> SET (p)`.

Example
```
> PARAMETER
estimate coefficients of regression dz=A+B*time to correct
flowmeter data
  >> SHIFT
     >>> NONE
         >>>> ANNOTATION: coefficient A (constant)
         >>>> INDEX     : 2 3 4  (identifies data sets)
         >>>> GUESS     : 4.0E-6 [kg/sec]
         <<<<
     <<<
  >> DRIFT
     >>> NONE
         >>>> ANNOTATION: coefficient B (slope)
         >>>> INDEX     : 2 3 4
         >>>> GUESS     : 1.0E-9 [kg/sec/sec]
         <<<<
     <<<
  <<
```

See Also
```
>> SCALING, >> LAG, >> SHIFT, >>> SET (p), >>>> INDEX (p)
```

Command
ECHO (OFF)

Parent Command
none

Subcommand
none

Description
Turns echo of interpreted commands and keywords on or off.

Example
```
> PARAMETER
estimate coefficients of regression dz=A+B*time to correct
flowmeter data

ECHO ON
  >> SHIFT
     >>> NONE
         >>>> ANNOTATION: coefficient A (constant)
         >>>> INDEX     : 2 3 4  (identifies data sets)
         >>>> GUESS     : 4.0E-6 [kg/sec]
         <<<<
     <<<
  >> DRIFT
     >>> NONE
         >>>> ANNOTATION: coefficient B (slope)
         >>>> INDEX     : 2 3 4
         >>>> GUESS     : 1.0E-9 [kg/sec/sec]
         <<<<
ECHO OFF
     <<<
  <<
```

See Also
-

Command

```
>>> ELEMENT                  : eleme (eleme_i ...) (+ iplus)
>>> ELEMENT COORDINATE       : X Y Z
>>> ELEMENT COORDINATE BOX : Xmin Ymin Zmin Xmax Ymax Zmax
>>> ELEMENT COORDINATE ELLIPSOID : Xc Yc Zc DX DY DZ
>>> ELEMENT COORDINATE CYLINDER : Xs Ys Zs Xe Ye Ze Radius
>>> ELEMENT COORDINATE CUBE: Xc Yc Zc DX DY DZ
>>> ELEMENT PROFILE/CROSS-SECTION/MAP
```

or:

```
>>> MATERIAL                : materials
```

Parent Command

any second-level command in block `> OBSERVATION` requiring element names.

Subcommand

any fourth-level command in block `> OBSERVATION`

Description

This command reads one or more element names, or coordinates or a coordinate range. Most observation types refer to a variable that is associated with a grid block (as opposed to a connection or sink/source name). Element names are designated by a three-character/two-integer (FORTRAN format: AAAII) code name. Blanks in the element names as printed in the TOUGH2 output file must be replaced by underscores (e.g. an element name specified in the TOUGH2 input file as '`B 007`' is printed as '`B 0 7`' in the TOUGH2 output file. Therefore, it has to be addressed in the iTOUGH2 input file as '`B_0_7`').
Multiple elements can be specified, and iTOUGH2 calculates the sum or mean of the corresponding output variable (see subcommands `>>>> SUM` and `>>>> MEAN`, respectively).
A sequence of *iplus* elements can be generated by increasing the number of the last element. The following two command lines are identical:

```
>>> ELEMENT: AA__1  BB_15  +3
>>> ELEMENT: AA__1  BB_15  BB_16  BB_17  BB_18
```

If keyword `COORDINATES` is present, coordinates can be specified, and the element closest to these coordinates will be identified.

If keywords `COORDINATE BOX`, `COORDINATE ELLISPODE`, `COORDINATE CYLINDER`, or `COORDINATE CUBE` are present, all elements within the specified region will be included in the element list. The region is defined according to the following table. If keyword `COMPLEMENT` or `OUTSIDE` is present, the complementary region is selected (i.e., all elements outside the defined geometry). Element coordinates must be present in columns 51–80 of the TOUGH2 `ELEME` block.

KEYWORD	*XREGION(i)*								
	1	2	3	4	5	6	7	8	9
(none)	X	Y	Z	-	-	-	-	-	-
BOX	X_{min}	Y_{min}	Z_{min}	X_{max}	Y_{max}	Z_{max}	aximuth	dip	plunge
ELLIPSOID	X_{center}	Y_{center}	Z_{center}	$X_{semi-axis}$	$Y_{semi-axis}$	$Z_{semi-axis}$	aximuth	dip	plunge
CYLINDER	X_{start}	Y_{start}	Z_{start}	X_{end}	Y_{end}	Z_{end}	radius	-	-
CUBE	X_{center}	Y_{center}	Z_{center}	$X_{half-length}$	$Z_{half-length}$	$Z_{half-length}$	aximuth	dip	plunge

Add keyword ANGLE or ROTATE if the region is not aligned with the coordinate axes, and provide azimuth, dip, and plunge correction angles.

Keywords CONSTANT (default), LINEAR, SPHERICAL, EXPONENTIAL, or RAMP may be added to weigh the contribution of each observation according to its distance *d* from the center of the region.

$$\omega = 1 \qquad \text{(constant)}$$

$$\omega(d) = 1 - d \qquad \text{(linear)}$$

$$\omega(d) = 1 - \left(\frac{3}{2} d - \frac{1}{2} d^3 \right) \qquad \text{(spherical)}$$

$$\omega(d) = a^{-d} \qquad \text{(exponential)}$$

$$\omega(d) = \min(1, (1 - d)/a) \qquad \text{(constant-linear ramp)}$$

If keyword PROFILE, CROSS-SECTION, or MAP is present, the data (which must be provided in an external file, see command >>>> DATA FILE: *filename*) are organized spatially by lists of coordinates followed by columns of measured values for each survey time. The data will be assigned to the nearest element (element coordinates must be present in columns 51–80 of the TOUGH2 ELEME block.). Keyword PROFILE supports one-dimensionl, vertical data profiles (i.e., Z-coordinates), whereas CROSS-SECTION and MAP both require X-, Y-, and Z-coordinates.

Profile data are given as follows:

```
NZ  NT
XCoord  YCoord
      Time(1)    Time(2)    ...  Time(NT)
Z(1)  Data(1,1)  Data(1,2)  ...  Data(1,NT)
Z(2)  Data(2,1)  Data(2,2)  ...  Data(2,NT)
...   ...        ...        ...  ...
Z(NZ) Data(NZ,1) Data(NZ,2) ...  Data(NZ,NT)
```

Two- or three-dimensional data are given as follows:

```
NP  NT
                    Time(1)    Time(2)    ...  Time(NT)
X(1)  Y(1)  Z(1)    Data(1,1)  Data(1,2)  ...  Data(1,NT)
X(2)  Y(2)  Z(2)    Data(2,1)  Data(2,2)  ...  Data(2,NT)
...   ...   ...     ...        ...        ...  ...
X(NP) Y(NP) Z(NP)   Data(NP,1) Data(NP,2) ...  Data(NP,NT)
```

If command `>>> MATERIAL` is used, all elements assigned to the specified material(s) will be selected.

Example

```
> OBSERVATION
  >> GAS SATURATION
     >>> ELEMENTS: ELM_0      + 99
         >>>> ANNOTATION    : Mean saturation
         >>>> take the VOLUMEtric MEAN of all 100 elements
         >>>> DATA on FILE  : Sg.dat
         >>>> DEVIATION     : 0.05
         <<<<
     <<<

  >> CAPILLARY pressure
     >>> ELEMENT COORDINATE BOX: 10. 0.4 -5.0 40.0 0.5 -1.0
         >>>> ANNOTATION    : Mean Pcap
         >>>> take the VOLUMEtric MEAN of all elements in box
         >>>> DATA on FILE  : Pc.dat
         >>>> DEVIATION     : 1.0E4
         <<<<
     <<<
  <<
```

See Also

`>>> CONNECTION, >>> SINK, >>>> MEAN, >>>> SUM`

Command

```
>>> EMPIRICAL (MATRIX: ndim) (iTOUGH2) (CORRELATION)
```

or

```
>>> EOF (MATRIX: ndim) (iTOUGH2) (CORRELATION)
```

Parent Command

```
>> ERROR
```

Subcommand

-

Description

This command must be used in combination with command `>>> MONTE CARLO` to invoke stochastic simulations of correlated parameters using Empirical Orthogonal Functions (EOF). EOF is a variant of Monte Carlo simulations to quantify the uncertainty of model predictions as a result of parameter uncertainty. Many parameter sets are generated, following the predefined covariance matrix $\mathbf{C}_{pp}$. The i th parameter set $\mathbf{Y}_i$ is obtained by linear combination of the eigenvectors $\mathbf{u}_k$ of $\mathbf{C}_{pp}$ and stochastic coefficients $\phi_k(\xi_i)$:

$$\mathbf{Y}_i = \sum_{k=1}^{n} \mathbf{u}_k \phi_k$$

$$\phi_k(\xi_i) = \xi_i \cdot \sqrt{a_k}$$

where ξ is a standard normal distributed random variable, and a_k is the k th eigenvalue of $\mathbf{C}_{pp}$. For more details, see *Kitterød and Gottschalk* [1997].

The elements of matrix $\mathbf{C}_{pp}$ can be supplied using one of the following options:

(1) Provide indices and elements of $\mathbf{C}_{pp}$. Example:

```
>>> EMIRICAL ORTHOGONAL FUNCTIONS
    1   1   0.80643E-04
    2   2   0.71921E-04
    2   1   0.64412E-04
```

Use keyword `CORRELATION` if off-diagonal term is correlation coefficient instead of covariance. Example:

```
>>> EMPIRICAL ORTHOGONAL FUNCTIONS, CORRELATION
    1  1  0.80643E-04
    2  2  0.71921E-04
    2  1  0.864
```

(2) Provide keyword `MATRIX`, followed by a colon and the dimension *ndim* of the square matrix $\mathbf{C}_{pp}$. The lower triangle of the covariance matrix is then provided on exactly *ndim* additional lines. If keyword `CORRELATION` is present, the off-diagonal terms represent correlation coefficients rather than covariances. Example:

```
>>> EOF, dimension of CORRELATION MATRIX: 4
    9.1234
   -0.67     0.00413
    0.80     0.213     1.3E-6
    0.50    -0.155     0.90     4.3E12
```

(3) If calculated during a previous iTOUGH2 inversion, the covariance matrix can be taken from the iTOUGH2 output file and directly copied after the command line. This option is invoked by keyword `iTOUGH2`. The matrix will be read by formatted input, so it is crucial that the correct format is maintained. If *ndim* is greater than 6, the matrix is split in multiple submatrices. All submatrices must be copied exactly as they were printed to the iTOUGH2 output file. Example:

```
>>> EOF, MATRIX of dim.: 3 in iTOUGH2 format
                log(abs. perm.)    POROSITY SAND    Gas entrapped
log(abs. perm.)       .80643E-04            .846           -.253
POROSITY SAND         .64412E-04       .71921E-04           -.500
Gas entrapped        -.52623E-05      -.98296E-05       .53843E-05
```

Example

```
> COMPUTATION
  >> STOP
     >>> number of Monte Carlo SIMULATIONS: 250
     <<<
  >> ERROR propagation analysis
     >>> MONTE CARLO simulations, SEED number: 777
     >>> EOF, dimension of CORRELATION MATRIX: 3
         9.1234
        -0.67     0.00413
         0.80     0.213     1.3E-6
     <<<
  <<
```

See Also

>>> FOSM, >>> LATIN HYPERCUBE SAMPLING, >>> MONTE CARLO

Command
`>> ENTHALPY`

Parent Command
`> PARAMETER`

Subcommand
`>>> SOURCE`

Description
This command selects as a parameter the fixed specific enthalpy of the injected fluid (TOUGH2 variable *EX*) or the time-dependent specific enthalpy of the produced or injected fluid (TOUGH2 variable *F3(L)*, *LTAB*>1 and *ITAB* non-blank). This parameter refers to a sink/source code name. Estimating a time-dependent enthalpy requires providing index *L* through command `>>>> INDEX`.
Note that enthalpy is also an observation type (see command `>> ENTHALPY (o)`).

Example
```
> PARAMETER
  >> specific ENTHALPY
     >>> SOURCE: INJ_1
         >>>> ANNOTATION: fixed enthalpy
         <<<<
     >>> SOURCE: INJ_2 INJ_5
         >>>> ANNOTATION: variable enthalpy
         >>>> VALUE
         >>>> INDEX     : 1 2 3 8 9 10
         <<<<
     <<<
  <<
```

See Also
`>> TIME (p), >> ENTHALPY (o)`

Command
>> ENTHALPY (*phase_name*/PHASE: *iphase*) (WELLHEAD)(CHANGE/DELTA)

Parent Command
> OBSERVATION

Subcommand
>>> SINK

Description
This command selects as an observation type the flowing enthalpy in a production well. This observation type refers to a sink code name. If multiple sinks are provided, the flowing enthalpy is weighted by the individual production rates. If a phase is selected either by specifying a valid *phase_name* or a phase number *iphase* or through command >>>> PHASE, only the flowing enthalpy of that phase is calculated.
Note that the enthalpy of the injected fluid can also be an unknown parameter to be estimated (see command >> ENTHALPY (p)).
If keyword WELLHEAD is present, the wellhead enthalpy as calculated by the wellbore simulator FloWell is returned.

Example
```
> OBSERVATION
  >> ENTHALPY
     >>> SINK: WEL_1 + 5
         >>>> ANNOTATION    :  Flowing enthalpy
         >>>> FACTOR        :  1000.0        [kJ/kg] - [J/kg]
         >>>> DATA from FILE:  enthalpy.dat  [HOUR]
         >>>> DEVIATION     :  10.0          [kJ/kg]
         <<<<
     <<<
  <<
<
```

See Also
>> ENTHALPY (p)

Command
>>> EOF (MATRIX: *ndim*) (iTOUGH2) (CORRELATION)

Parent Command
>> ERROR

Subcommand
-

Description
(synonym for command >>> EMPIRICAL ORTHOGONAL FUNCTIONS)

Example
(see command >>> EMPIRICAL ORTHOGONAL FUNCTIONS)

See Also
>>> EMPIRICAL ORTHOGONAL FUNCTIONS

Command
```
>> ERROR
```

Parent Command
```
> COMPUTATION
```

Subcommand
```
>>> ALPHA
>>> EMPIRICAL ORTHOGONAL FUNCTIONS
>>> FISHER
>>> FOSM
>>> HESSIAN
>>> LATIN HYPERCUBE SAMPLING
>>> LINEARITY
>>> LIST
>>> MONTE CARLO
>>> POSTERIORI
>>> PRIORI
>>> TAU
```

Description
This is the parent command of a number of subcommands that deal with the *a posteriori* error analysis or uncertainty propagation analysis.

Example
```
> COMPUTATION
  >> ERROR analysis
     >>> use confidence level 1-ALPHA:= 95 %
     >>> use sigma as determined by FISHER model test
     >>> calculate finite difference HESSIAN matrix
     >>> check LINEARITY assumption on : 99 % level
     <<<
  <<
```

See Also
-

Command
>>>> EXECUTABLE: *FILE* (BEFORE/AFTER)

Parent Command
>>> PEST

Subcommand
-

Description
iTOUGH2 capabilities can be applied to non-TOUGH2 models using the PEST interface [*Doherty*, 2002]. The user-supplied model must be an executable that can be run from a system command prompt. The system command can also be a script of batch file, in which multiple models can be combined. The model must be able to be executed without user interference. It must get its input through one or multiple ASCII text files, and write its output to one or multiple ASCII text files. Communication between itough2 and the model's input and output occurs through PEST-style template and instruction files, respectively.

The user-supplied model can be run by itself, before (default) or after (keyword AFTER) a TOUGH2 run. The latter two options are useful for estimating parameters that relate to pre- or postprocessors of TOUGH2, respectively. The name of the executable, script, or batch file is provided following the colon. If the executable name contains spaces, the command line must be in quotes. Under Unix, it is recommended to add the keyword FILE to the command line or on a separate line, so the executable is automatically copied to the temporary directory. Examples include:

```
>>>> EXECUTABLE      : myModel.exe
>>>> EXECUTABLE      : Run-ModelA-and-ModelB.bat
>>>> EXECUTABLE FILE : UnixScript.sh
>>>> EXECUTABLE      : "a.out < input > output"
                FILE : a.out
```

Example
```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE FILE : this-is-not-TOUGH.exe
         >>>> TEMPLATE        : 1
              input.tpl          input.txt
         >>>> INSTRUCTION     : 1
              output.ins         output.txt
```

See Also
>>>> INSTRUCTION, >>>> TEMPLATE

Command
`>>>> FACTOR`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
The parameter to be estimated is a factor with which the initial TOUGH2 parameter value is multiplied:

$$p = X/X_0 \Leftrightarrow X = p \cdot X_0$$

Here, p is the estimated parameter, X is the TOUGH2 parameter, and X_0 is the initial value of the TOUGH2 parameter. This option is useful to estimate a scaling factor for variable initial and boundary conditions, or to determine the mean of a quantity while maintaining ratios (e.g., if estimating a common factor applied to all three permeability values in a model domain, the anisotropy ratio remains constant). If the factor is log-normally distributed, add command `>>>> LOGARITHM` or use command `>>>> LOG(F)`. Estimating a factor is opposed to estimating the parameter value directly (command `>>>> VALUE`) or its logarithm (command `>>>> LOGARITHM (p)`).

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: CLAY1 SAND1
         >>>> estimate multiplication FACTOR, and maintain
              both the anisotropy ratio within a layer as
              well as the permeability ratio between clay
              and sand.
         >>>> INDEX        : 1 2 3
         >>>> initial GUESS: 1.0  (default)
         >>>> RANGE        : 0.01 100.0
         <<<<
     <<<
  <<
```

See Also
`>>>> LOGARITHM (p), >>>> LOG(F), >>>> VALUE`

Command
>>>> FACTOR: *factor*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command provides a conversion factor with which the data are multiplied so they comply with standard TOUGH2 units. The conversion factor is also applied to the standard deviation (see command >>>> DEVIATION (o)), i.e., the measurement error must be given in the same units as the data. The standard TOUGH2 units are used throughout the iTOUGH2 output file, i.e., all observed and calculated values as well as residuals and standard deviations have been multiplied by *factor*.

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: WEL99
         >>>> conversion FACTOR: 1E5   [bar] - [Pa]
         >>>> DATA in HOURS and bar on FILE: pres.dat
         >>>> standard DEVIATION: 0.01 [bar]
         <<<<
     <<<
  <<
```

See Also
>> SCALING

Command
```
>>> FISHER
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
The *a posteriori* estimated error variance s_0^2 represents the variance of the mean weighted residual and is thus a measure of goodness-of-fit:

$$s_0^2 = \frac{\mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r}}{m-n}$$

The value s_0^2 is used in the subsequent error analysis. For example, the covariance matrix of the estimated parameters, $\mathbf{C}_{pp}$, is directly proportional to the scalar s_0^2. Note that if the residuals are consistent with the distributional assumption about the measurement errors (i.e., matrix $\mathbf{C}_{zz}$), then the estimated error variance assumes a value close to one. s_0^2 is also an estimate for the true or *a priori* error variance σ_0^2. It can be shown that the ratio s_0^2/σ_0^2 follows an F-distribution with the two degrees of freedom $f_1 = m-n$, and $f_2 = \infty$. Therefore, it can be statistically tested to see whether the final match deviates significantly from the modeler's expectations, expressed by matrix $\mathbf{C}_{zz}$. This is called the Fisher Model Test. The user must decide whether the error analysis should be based on the *a posteriori* or *a priori* error variance (see commands `>>> POSTERIOR` and `>>> PRIOR`, respectively). The decision can also be delegated to the Fisher Model Test according to the following table:

Fisher Model Test	Error Variance	Comment
$s_0^2/\sigma_0^2 > F_{m-n,\infty,1-\alpha}$	s_0^2	Error in functional and/or stochastic model
$1 \le s_0^2/\sigma_0^2 \le F_{m-n,\infty,1-\alpha}$	s_0^2	Model test passed
$s_0^2/\sigma_0^2 < 1$	σ_0^2	Error in stochastic model

Example
```
> COMPUTATION
  >> ERROR
     >>> FISHER model test decides whether the a priori or a
         posteriori error variance should be used
     <<<
```

See Also
```
>>> ALPHA, >>> POSTERIORI, >>> PRIORI
```

Command

```
>> FLOW (phase_name/PHASE: iphase) (component_name/COMPONENT:
icomponent (HEAT)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> CONNECTION
```

Description

This command selects as an observation type the total fluid flow rate, flow of a given phase, total component rate, rate of component in a given phase, or total heat flow rate. This observation type refers to a connection.

The phase name *phase_name* or phase number *iphase*, and the component name *component_name* or component number *icomponent*, which depend on the EOS module being used, are listed in the iTOUGH2 header. They can be specified either on the command line or using the subcommands `>>>> PHASE` or `>>>> COMPONENT`. If no phase is specified, the total flow rate is selected; if no component is specified, all the components in the specified phase are evaluated. If keyword `HEAT` is present, the total heat flux across the connection is selected. Note that the sign of the calculated flow rate depends on the order of the two elements in a connection.

Example

```
> OBSERVATION

  >> FLOW rate
     >>> CONNECTION: A11_1 A11_2  B11_1 B11_2  C11_1 C11_2 &
                      D11_1 D11_2  E11_1 E11_2  F11_1 F11_2
         >>>> ANNOTATION: Vapor flow across boundary
         >>>> Take WATER COMPONENT
         >>>> in GAS PHASE
         >>>> then take the SUM
         >>>> of the ABSOLUTE values of the 6 flow rates
         >>>> DATA in [kg/s] and MINUTES on FILE: outflow.dat
         >>>> standard DEVIATION: 0.1 kg/sec
         <<<<
     <<<
  <<
```

See Also

-

Command
```
>> FORCHHEIMER
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects a parameter of the non-Darcy flow coefficient model (TOUGH2 variable *FORCH(IPAR)*). Use command `>>>> INDEX` to select the parameter index *IPAR*.

Example
```
> PARAMETER
  >> parameter of FORCHHEIMER non-Darcy flow model
     >>> MODEL
         >>>> ANNOTATION    : Constant Beta
         >>>> INDEX         : 1
         >>>> LOGARITHM
         >>>> RANGE         : 3.0 8.0
         <<<<
     <<<
  <<
```

See Also
-

Command
>>> FORMAT: *format* (LIST)
or
>>> PLOTFILE: *format* (LIST)

Parent Command
>> OUTPUT

Subcommand
-

Description
iTOUGH2 does not directly generate graphs (except for the residual plot and correlation chart). However, it does generate a plot file with data at the calibration points and system response as calculated with the initial, intermediate (see command >>> PLOTTING), and final parameter set. A plot file with the relative permeability and capillary pressure curves can also be requested (see command >>> CHARACTERISTIC).
These plot files must be processed by an external visualization package.
iTOUGH2 generates plot files in PLOPO format, a visualization software developed by U. Kuhlmann at ETH, ZŸrich. The PLOPO plot file is internally reformatted to comply with formats of other visualization programs. The string *format* identifies the visualization software (for a list of available formats add keyword LIST). The reformatted plot files have a file extension specific for the chosen software (for example "*.tec*" for TECPLOT.)
A general format accepted by most commercially available plotting programs is the arrangement in columns, where the first column contains the time, and additional columns hold the data and calculated system response for various observations. This general format can be obtained by using keyword COLUMN.
The default format is TECPLOT, and can be changed to another format in file *it2main.f*, BLOCK DATA IT, through variable *IPLOTFMT*.
Additional interfaces can be programmed into subroutine PLOTIF and REFORMAT.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> FORMAT of plot file: COLUMNS (print LIST of
         available formats)
     <<<
  <<
```

See Also
>>> CHARACTERISTIC, >>> PLOTTING

Command
>>>> FORMAT: *format*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command accepts a FORTRAN format statement for reading data. By default, data are read in free format following the >>>> DATA command or directly from a data file. If a column contains non-numeric characters, formatted input can be invoked by providing a string *format* representing the corresponding FORTRAN format statement. The format statement must be in brackets and must not contain any blanks.

Example
```
> OBSERVATION
  >> LIQUID FLOW rate
     >>> CONNECTION : ELM50  BOT_1
         >>>> multiply measurements by FACTOR: -1.0E-06
         >>>> FORMAT: (3X,F11.1,30X,E17.10)
                        used to read time and value
         >>>> COLUMN: 1 2 (time in col. 1, flow rate col. 2
                            of above FORMAT statement)
         >>>> HEADER:   3 lines before values
         >>>> DATA in MINUTES
------------------------------------------------------------
  # Time [min]  Cap. Pres, [hPa] FlowmeterX Flow Rate [mg/s]
------------------------------------------------------------
  1       5.0   0.1698331055E+02 FlowmeterA 0.9869399946E+01
  2      10.0   0.2075763428E+02 FlowmeterA 0.1039689596E+02
  3      15.0   0.2357142822E+02 FlowmeterA 0.1162893932E+02
  4      20.0   0.2529052490E+02 FlowmeterA 0.1353439620E+02
  5      25.0   0.2598133789E+02 FlowmeterA 0.1469761628E+02
  .      ....   ................ ......... ................

         >>>> RELATIVE error is: 10.0 % of
              the individual measurement
         <<<<
     <<<
```

See Also
>>>> DATA, >>>> COLUMN, >>>> HEADER, >>>> PICK

Command
```
>>> FORWARD
```
or
```
>>> DIRECT
```

Parent Command
```
>> CONVERGE
```
or
```
>> OPTION
```

Subcommand
-

Description
This command allows one TOUGH2 simulation to be performed in order to solve the forward problem.
It is advantageous to perform a single TOUGH2 simulation before invoking more expensive inversions. The result from the forward run can be used to check TOUGH2 and iTOUGH2 input. Furthermore, a plotfile is generated with the results from the simulation with the initial parameter set, which can be compared to the observed data. The CPU time requirement for an inversion can also be estimated from a single TOUGH2 simulation.
(Note that there is another command `>>> FORWARD` in block `>> JACOBIAN`.)

Example
```
> COMPUTATION
  >> STOP after
     >>> solving the DIRECT problem

#    >>> before performing : 5 iTOUGH2 ITERATIONS

     <<<
  <<
```

See Also
```
>>> SIMULATION
```

Command
```
>>> FORWARD (: iswitch)
```

Parent Command
```
>> JACOBIAN
```

Subcommand
-

Description
With this command the elements of Jacobian matrix are calculated by means of a forward finite difference quotient:

$$J_{ij} = \frac{\partial z_i}{\partial p_j} \approx \frac{z_i(\mathbf{p}; p_j + \delta p_j) - z_i(\mathbf{p})}{\delta p_j}$$

The evaluation of the Jacobian thus requires n+1 TOUGH2 simulations, where n is the number of parameters. The size of the perturbation δp_j can be controlled using command `>>> PERTURB`.
The Jacobian is used in both the minimization algorithm and the *a posteriori* error analysis. For the Levenberg-Marquardt minimization algorithm, the accuracy obtained by using a forward (as opposed to centered) finite difference quotient is usually sufficient, especially during the first few iterations far away from the minimum. However, if approaching the minimum and especially for the subsequent error analysis one might want to use a more accurate approximation of the Jacobian. If a colon is given on the command line followed by an integer *`iswitch`*, iTOUGH2 switches from forward to centered finite differences after *`iswitch`* iterations.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> perform a total of : 6 iTOUGH2 ITERATIONS
     <<<
  >> JACOBIAN
     >>> use FORWARD finite differences for : 5 iterations,
         i.e. a centered finite difference quotient is used
         for the final iteration and error analysis.
     >>> PERTURBation factor is : 0.005 times the
         parameter value
     <<<
  <<
```

See Also
```
>>> CENTERED, >>> PERTURB
```

Command
```
>>> FOSM (MATRIX: ndim) (iTOUGH2) (CORRELATION) (DIAGONAL)
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
This command performs First-Order-Second-Moment (FOSM) uncertainty propagation analysis. FOSM quantifies the uncertainty of model predictions as result of parameter uncertainty. FOSM is the analysis of the mean and covariance of a random function based on its first order Taylor series expansion. FOSM analysis presumes that the mean and covariance are sufficient to characterize the distribution of the dependent variables, i.e., the model results are assumed to be normally distributed, and perturbations about the mean can be approximated by linear functions $\mathbf{J}$. The covariance of the uncertain parameters, $\mathbf{C}_{pp}$, is translated into the covariance of the simulated system response, $\mathbf{C}_{\hat{z}\hat{z}}$:

$$\mathbf{C}_{\hat{z}\hat{z}} = \mathbf{J}\mathbf{C}_{pp}\mathbf{J}^T$$

The diagonal elements of matrix $\mathbf{C}_{pp}$, i.e., the variances of the parameters, can be supplied by command `>>>> VARIANCE` (or related commands) in block `> PARAMETER`. If correlations are to be taken into account, the full covariance matrix must be provided. The elements of matrix $\mathbf{C}_{pp}$ can be supplied using one of the following options:

(1) Provide indices and elements of $\mathbf{C}_{pp}$. Example:
```
 >>> FOSM error analysis
     1  1  0.80643E-04
     2  2  0.71921E-04
     2  1  0.64412E-04
```

Use keyword `CORRELATION` if off-diagonal term is correlation coefficient instead of covariance. Example:
```
 >>> FOSM with CORRELATION coefficients provided
     1  1  0.80643E-04
     2  2  0.71921E-04
     2  1  0.864
```

(2) Provide keyword `MATRIX`, followed by a colon and the dimension *ndim* of the square matrix $\mathbf{C}_{pp}$. The lower triangle of the covariance matrix is then provided on exactly *ndim* additional lines. If keyword `CORRELATION` is present, the off-diagonal terms represent correlation coefficients rather than covariances. Example:

```
>>> FOSM, dimension of CORRELATION MATRIX: 4
    9.1234
   -0.67     0.00413
    0.80     0.213     1.3E-6
    0.50    -0.155     0.90     4.3E12
```

(3) If calculated during a previous iTOUGH2 inversion, the covariance matrix can be taken from the iTOUGH2 output file and directly copied after the command line. This option is invoked by keyword `iTOUGH2`. The matrix will be read by formatted input, so it is crucial that the correct format is maintained. If *ndim* is greater than 6, the matrix is split in multiple submatrices. All submatrices must be copied exactly as they were printed to the iTOUGH2 output file. Example:

```
>>> FOSM, read MATRIX of dim.: 3 in iTOUGH2 format
                log(abs. perm.)    POROSITY SAND    Gas entrapped
log(abs. perm.)         .80643E-04             .846           -.253
POROSITY SAND           .64412E-04       .71921E-04           -.500
Gas entrapped          -.52623E-05      -.98296E-05       .53843E-05
```

If the full matrix is provided, but only the diagonal terms (variances) shall be used in the error analysis, use keyword `DIAGONAL` on the command line. This option makes it easy to study the impact of correlations on the uncertainty propagation analysis. The uncertainty of the model prediction as a result of parameter uncertainty is given as a standard deviation in the residual analysis. Furthermore, the plotfile contains the system response for the mean parameter set as well as error band on the specified confidence level (see command `>>> ALPHA`). It is suggested to also increase the perturbation factor for calculating the Jacobian matrix, and to use a centered finite difference quotient. This yields in a more realistic the error band if the model is highly non-linear. It should be realized, however, that Monte Carlo is the preferred method if dealing with highly non-linear flow systems.

Example

```
> COMPUTATION
  >> ERROR propagation analysis
     >>> perform First-Order-Second-Moment (FOSM) analysis
     >>> error bands on (1-ALPHA)=: 95 % confidence level
     <<<
  >> JACOBIAN
     >>> use CENTERED finite difference quotient
     >>> PERTURBATION factor at least: 5.0 %
     <<<
  <<
```

See Also

`>>> ALPHA, >>> EMPIRICAL ORTHOGONAL FUNCTIONS, >>> LATIN HYPERCUBE, >>> MONTE CARLO`

Command
```
>>>> GAUSSIAN
```
or
```
>>>> NORMAL
```

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command generates normally distributed input parameters for Monte Carlo simulations. This is the default distribution. Parameter values will be generated following a normal distribution with the initial guess as the mean, and the standard deviation taken from command `>>>> DEVIATION`. Only values within the specified range will be accepted. If command `>>>> LOGARITHM` is also present, parameters follow a log-normal distribution.

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: SAND1 BOUND WELLB
         >>>> ANNOTATION         :  log(k) is uncertain
         >>>> LOGARITHM
         >>>> generate log-NORMAL distribution about mean...
         >>>> initial GUESS      : -12.0         and with...
         >>>> standard DEVIATION : 1.0         within the...
         >>>> admissible RANGE   : -15.0 -9.0
         <<<<
     <<<
  <<

> COMPUTATION
  >> perform ERROR propagation analysis by means of...
     >>> : 1000 MONTE CARLO simulations using...
     >>> LATIN HYPERCUBE sampling
     <<<
  <<
```

See Also
```
>>> LATIN HYPERCUBE, >>> MONTE CARLO, >>>> UNIFORM,
>>>> TRIANGULAR
```

Command
```
>>> GAUSS-NEWTON
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command performs Gauss-Newton steps to minimize the objective function. The Gauss-Newton algorithm assumes linearity and can be described as follows:

$$\Delta \mathbf{p} = \left(\mathbf{J}^T \mathbf{C}_{zz}^{-1} \mathbf{J} \right)^{-1} \mathbf{J}^T \mathbf{C}_{zz}^{-1} \mathbf{r}$$

$$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \Delta \mathbf{p}$$

Gauss-Newton steps are efficient if the model is linear (only one iteration required to find minimum) or nearly linear. If the model is highly nonlinear, Gauss-Newton steps are usually too large, leading to an inefficient or even unsuccessful step.
By default, iTOUGH2 uses the Levenberg-Marquardt minimization algorithm, which is a modification of the Gauss-Newton algorithm.

Example
```
> COMPUTATION
  >> OPTION
     >>> use GAUSS-NEWTON minimization algorithm
     <<<

  >> STOP
     after >>> :1 ITERATION
     <<<
  <<
```

See Also
```
>>> ANNEAL, >>> GRID SEARCH, >>> LEVENBERG-MARQUARDT,
>>> SIMPLEX
```

Command

```
>> GENERATION (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
```

or

```
>> PRODUCTION (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> SINK
```

Description

This command selects as an observation type the total or fractional generation rate in a production well. This observation type refers to a sink code name.

The total generation rate is usually prescribed in TOUGH2 block GENER, or can be considered a parameter to be estimated (see command `>> RATE`). A variable generation rate suitable for calibration is obtained only for wells on deliverability (type DELV), or if the generation rate of a specific phase or component is used.

The fractional generation rate refers to the production of a specific phase or component. The component name *comp_name* or component number *icomp*, as well as the phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the iTOUGH2 header. They can be specified either on the command line or using the subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively. If neither a phase nor a component is specified, the total generation rate is calculated. If a phase but no component is selected, the generation rate of that phase including all components is calculated. If a component but no phase is selected, the generation rate of that component in all phases is calculated. Finally, if a specific phase and a specific component is given, only the generation rate of that component in the specified phase is calculated.

Example

```
> OBSERVATION
  >> GAS GENERATION (includes both air and vapor)
     >>> SINK: DLV_1 + 5
         >>>> FACTOR         :  1000.0        [ml/s] - [kg/s]
         >>>> DATA on FILE   :  gasflow.dat   [HOUR]
         >>>> RELATIVE error:  10.0          [%]
         <<<<
     <<<
  <<
```

See Also

```
>> CUMULATIVE, >> RATE, >> TOTAL MASS, >>>> COMPONENT,
>>>> PHASE
```

Command
```
>>> GOODNESS-OF-FIT
```

Parent Command
```
>> STOP
```

Subcommand
-

Description
Optimization using the Levenberg-Marquardt minimization algorithm shall be stopped of the match reaches the expected goodness-of-fit, i.e., if the estimated error variance is less than 1.0.

Example
```
> COMPUTATION
  >> STOP
     >>> GOODNESS-OF-FIT
     <<<
```

See Also
-

Command
`>>> GRID BLOCK: eleme (eleme_i ...) (+ iplus)`

Parent Command
any second-level command in block `> OBSERVATION` requiring element names.

Subcommand
any fourth-level command in block `> OBSERVATION`

Description
(synonym for command `>>> ELEMENT`)

Example
(see command `>>> ELEMENT`)

See Also
`>>> ELEMENT`

Command

```
>>> GRID SEARCH (: ninval1 (ninval2 (inval3)) /
                  FILE: filename) (UNSORTED)
```

or

```
>>> OBJECTIVE (: ninval1 (ninval2 (inval3)) /
                FILE: filename) (UNSORTED)
```

Parent Command

```
>> OPTION
```

Subcommand

-

Description

This command evaluates the objective function for a number of specific parameter sets. A arbitrarily long list of parameter sets can be provided on file *filename*, each row consisting of one parameter set, whereby inactive and tied parameters are to be omitted. Alternatively, parameter sets are internally generated on a regular grid, mapping out the entire parameter space. In this case, lower and upper bounds must be defined for each parameter using command `>>>> RANGE` in block `> PARAMETER`. This range is then subdivided into *ninteri* intervals by inserting *invali*+1 equally spaced points, generating parameter sets on a regular grid in the *n*-dimensional parameter space. The objective function is evaluated at each grid point. Keyword `UNSORTED` may be used in PVM applications to improve efficiency. Evaluating the objective function throughout the entire parameter space is referred to as grid searching. The parameter set, the value of the objective function, and the contribution of each observation type to the objective function is printed to the iTOUGH2 output file. The global minimum is likely to be in the vicinity of the parameter set with the lowest objective function. Furthermore, the information listed in the output file can be used to visually represent and study the topology of the objective function. For example, one can generate a contour plot of the objective function for *n*=2, which may reveal the presence local minima. If only one output variable is defined in block `> OBSERVATION` and no data are provided, this option can also be used in combination with command `>>> L1-ESTIMATOR` to examine the sensitivity of the output variable over an extensive parameter range.

Example

```
> COMPUTATION
  >> OPTION
     >>> GRID SEARCH: 20 10 intervals along parameter axes
     <<<
   <<
```

See Also

`>>> ANNEAL`, `>>> GAUSS-NEWTON`, `>>> LEVENBERG-MARQUARDT`, `>>> PVM`, `>>> SIMPLEX`, `>>> TORNADO`

Command
>> GUESS (FILE: *file_name*)

Parent Command
> PARAMETER

Subcommand
-

Description
This command identifies initial guesses of the parameters to be estimated. The initial guess vector $\mathbf{p}^{(k=0)} = \mathbf{p}_0$ is the starting point of the minimization algorithm (iteration $k = 0$). Usually, the initial guess vector is identical with vector $\mathbf{p}^*$ which holds prior information about the parameters. By default, $\mathbf{p}_0$ and $\mathbf{p}^*$ are identical, taken from the TOUGH2 input file, and overwritten by command >>>> PRIOR. The starting point for the minimization algorithm may be different from the prior information vector. In this case, prior information is taken from the TOUGH2 input file or must be provided through command >>>> PRIOR. The starting point is taken from command >> GUESS which overwrites the initial guess provided through command >>>> GUESS.

A parameter is identified by an integer value indicating its position in the parameter block:
I XIGUESS(I)

This format is identical to that of file *<invfile>.par*. Initial guesses can be provided either following the command line, or read from a file if keyword FILE is present. The filename is given after the colon. This latter option is useful to transfer best estimates from one inversion to another if the order of the parameters is the same.

Example
```
> PARAMETER
  >> GUESS
  2   0.345   (initial guess for parameter no. 2)
  3 -16.971   (initial guess for parameter no. 3)

/* (the initial guess for parameter no. 1 is taken from the
    fourth-level command >>>> GUESS, or - if not present -
    from the TOUGH2 input file) */

> PARAMETER
  >> read GUESS from FILE: testi.par
```

See Also
>>>> GUESS, >>>> PRIOR

Command
>>>> GUESS: *guess*

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command provides an initial guess of the parameter to be estimated. If neither this command nor command >> GUESS is used, the initial guess is taken from the TOUGH2 input file. The initial guess is the starting point for the minimization algorithm, to be distinguished from prior information (see command >>>> PRIOR). The initial guess can be overwritten by the second-level command >> GUESS.
If command >>>> LOGARITHM is present, the initial guess is the logarithm of the parameter. Similarly, if command >>>> FACTOR is present, the initial guess should be a multiplication factor (default is 1.0).

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> ROCK types: CLAY1 CLAY2 CLAY3 BOUND
         >>>> FACTOR
         >>>> initial GUESS     :   1.0
         >>>> is not WEIGHTed   :   0.0  (default)
         <<<<
     >>> ROCK type : SAND1
         >>>> LOGARITHM
         >>>> PRIOR information : -12.0
         >>>> standard DEVIATION:   1.0 order of magnitude
         <<<<
     <<<

  >> GUESS, i.e., starting point for optimization
     2  -13.0
  <<
```

See Also
>> GUESS, >>>> PRIOR, >>>> DEVIATION, >>>> VARIATION

Command
>>>> HEADER: *nskip*
or
>>>> SKIP: *nskip*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command identifies the number of lines to be skipped before reading data. Data reading starts *nskip*+1 lines after the command line >>>> DATA, or on the (*nskip*+1)-th line of the data file (see command >>>> DATA). Lines containing data are skipped using command >>>> PICK.

Example
```
> OBSERVATION
  >> TEMPERATURE
     >>> ELEMENTS: ELM_10 + 4
         >>>> SKIP: 3 lines before reading data
         >>>> DATA  [HOURS]
 (1) ------------------------------------------------------
 (2)    time [h]   temperature   comment
 (3) ------------------------------------------------------
         0.00          21.3   mean temperature prior to test
         1.12          21.3   heater turned on
         1.15          21.9
         1.20          23.4
         2.00          32.8   heater turned off
         2.10          29.3
         2.20          26.7
         2.30          24.1
         4.00          21.6   end of experiment
     ------------------------------------------------------
         >>>> standard DEVIATION: 0.5 degrees C
         <<<<
     <<<
  <<
```

See Also
>>>> COLUMN, >>>> DATA, >>>> FORMAT, >>>> PICK, >>>> SET (o)

Command
HELP (a keyword in combination with any command)

Parent Command
-

Subcommand
-

Description
A short message about the command usage is printed to the iTOUGH2 output file if keyword HELP is present on the command line. See also LIST and >>> INDEX for further support. If command >>> INPUT is used, the help message can be retrieved without performing any iTOUGH2 calculations.

Example
```
> COMPUTATION
  >> CONVERGE (what does this command do? HELP!)
     >>> print a LIST of available commands,
     >>> then stop after INPUT is read (HELP again!)
     <<<
  <<
```

See Also
>>> INDEX, LIST

Command
```
>>> HESSIAN
```

Parent Command
```
>> ERROR
>> JACOBIAN
```

Subcommand
-

Description
This command computes a finite difference Hessian matrix **H** for the error analysis following optimization. The elements of **H** are given by:

$$H_{jk} = 2\sum_{i=1}^{m} \frac{1}{\sigma_i^2}\left[\frac{\partial z_i}{\partial p_j}\frac{\partial z_i}{\partial p_k} - r_i \frac{\partial^2 z_i}{\partial p_j \partial p_k}\right]$$

The evaluation of **H** by means of finite differences requires $2n + n(n-1)/2$ additional TOUGH2 simulations, where n is the number of parameters. By default, the Hessian matrix, which is the inverse of the parameter covariance matrix, is approximated by $\mathbf{J}^T\mathbf{C}_{zz}^{-1}\mathbf{J}$ based on the linearity assumption, i.e., the second derivative term is ignored. Evaluating the finite difference Hessian, which takes into account the non-linearities, provides a means by which to check the linearity assumption (for another approach see command `>>> LINEARITY`). This may lead to a more accurate calculation of the covariance matrix of the estimated parameters. However, inclusion of the second-derivative term may yield a Hessian matrix that is not positive definite due to the presence of outliers, strong non-linearities, or the fact that the minimum has not been detected accurately, i.e., when the positive and negative residuals r_i do not cancel each other. In this case, iTOUGH2 automatically proceeds with the linearized Hessian which is positive definite by definition.

Example
```
> COMPUTATION
  >> ERROR analysis should be based on
     >>> finite difference approximation of HESSIAN matrix
     <<<
```

See Also
```
>>> LINEARITY
```

Command
```
>> HUMIDITY
```

Parent Command
```
> OBSERVATION
```

Subcommand
```
>>> ELEMENT
```

Description
This command selects relative humidity (a dimensionless fraction between 0 and 1) as an observation type. This observation type refers to an element.

Example
```
> OBSERVATION
  >> relative HUMIDITY
     >>> ELEMENT       : TCP_1
         >>>> FACTOR   :   0.01
         >>>> humidity DATA in % on file: RH.dat
         >>>> DEVIATION: 2.0 percent
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>> IDENTIFIABILITY
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
Performs and prints results of parameter identifiability analysis [*Doherty and Hunt*, 2009]. Parameter identifiability is a measure that can be used to rank input parameters in terms of their relative identifiability based on a calibration dataset. Identifiability is defined as the capability of model calibration to constrain parameters used by a model. It requires that the sensitivity of each model parameter be calculated for each observation. Singular value decomposition (SVD) of the weighted sensitivity matrix is then undertaken to quantify the relation between the parameters and observations that, in turn, allows selection of calibration solution and null spaces spanned by unit orthogonal vectors. Parameter identifiability is quantitatively defined as the direction cosine between a parameter and its projection onto the calibration solution space. This varies between zero and one, with zero indicating complete non-identifiability and one indicating complete identifiability.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print IDENTIABILITY matrices
     <<<
  <<
```

See Also
```
>>> SENSITIVITY
```

Command
```
>> IFS
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects as a parameter one of the IFS parameters for describing heterogeneity (see *Doughty* [1995]). An index `I` must be provided through command `>>>> INDEX`. If *I* is positive, the parameter is one of the affine transform entries (variable *PIFS(I)*). If *I* is negative, the parameter is the increment parameter for property field *-I* (variable *TINC(-I)*). If *I* is lower than -100, the parameter is the smoothing parameter in direction (*-I*-100) (variable *SMOOTH(-I*-100*)*).

Example
```
> PARAMETER
  >> IFS parameter
     >>> MODEL
         >>>> ANNOTATION: Diag. elements of B
         >>>> INDEX     : 1 4
         >>>> LOGARITHM
         <<<<
     >>> MODEL
         >>>> ANNOTATION: Increment
         >>>> INDEX     : -1
         >>>> VALUE
         <<<<
     >>> MODEL
         >>>> ANNOTATION: Smoothing in X- and Y direction
         >>>> INDEX     : -101 -102
         >>>> FACTOR
         <<<<
     <<<
  <<
```

See Also
-

Command
`>>>> IMMOBILIZATION (: ofredmin)`

Parent Command
`>>> SELECT`

Subcommand
-

Description
Employs "Temporary Parameter Immobilization" [*Skahill and Doherty*, 2006] (also referred to as "Automatic User Intervention") if the Levenberg-Marquardt algorithm is used. Insensitive parameters are fixed if the reduction in the objective function is less than *ofredmin* (default = 10%). If *ofredmin* is negative, insensitive parameters are fixed if the objective function is increased by less than –*ofredmin*; however, any trial parameter set that reduces the objective function is accepted.

Example
```
> COMPUTATION
  >> OPTION
     >>> SELECT parameters
         >>>> IMMOBILIZE if OF reduction is less than: 20 %
         <<<<
      <<<
  <<
```

See Also
-

Command
`>>>> INACTIVE`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command makes a parameter inactive. The initial guess provided for the corresponding parameter is transferred to TOUGH2, but the parameter is not evaluated or used for sensitivity analyses, parameter estimation, or uncertainty propagation analysis.

Example
```
> PARAMETER
  >> GENERATION rate
     >>> SOURCE: INJ_1 + 9
         >>>> ANNOTATION: early-time injection
         >>>> INDEX: 1 2 3 4 5
         >>>> FACTOR
         >>>> GUESS: 2.0
         >>>> INACTIVE
         <<<<
      <<<
  <<
```

See Also
-

Command
`INCLUDE FILE: file_name`

Parent Command
none

Subcommand
none

Description
Includes contents of file *file_name*. Note that the included file must contain valid iTOUGH2 commands on the appropriate command level.

There is no limit to the number of files that can be included. Included files can include other files, up to five levels.

Example

```
INCLUDE FILE: problem.par
INCLUDE FILE: problem.obs
> COMPUTATION
  >> STOP
     >>> ITERATION: 10
     <<<
  <<
<
```

See Also
-

Command
>>> INCOMPLETE: *max_incomplete*

Parent Command
>> CONVERGE

Subcommand
-

Description
A successful iTOUGH2 run is based on the robustness and stability of the underlying TOUGH2 simulation. It is therefore imperative to develop a TOUGH2 model that is capable of completing the desired simulation for a variety of parameter combinations. The simulation must reach the time of the last calibration point, i.e., a premature termination due to convergence failures is not acceptable. The number of potential convergence failures or errors leading to premature termination is large. The type of convergence failure is indicated in the iTOUGH2 output file. It is usually impossible to continue the optimization process after a convergence failure. However, in some cases iTOUGH2 is able to retrieve information from a previous simulation, which allows it to continue the inversion despite an incomplete run. iTOUGH2 always terminates if an incomplete run is encountered during the first evaluation of the Jacobian. The maximum number of incomplete simulations to be accepted by iTOUGH2 can be set by variable *max_incomplete* (default: 5). However, one should not rely on inversions that contain incomplete TOUGH2 runs.
Note that if option >>> STEADY-STATE is invoked, all incomplete simulations are accepted by iTOUGH2 assuming that steady-state conditions have been reached.

Example
```
> COMPUTATION
  >> TOLERANCE
     >>> perform    : 10 ITERATIONS
     >>> or stop if : 250 TOUGH2 SIMULATIONS are executed
     >>> accept     : 10 INCOMPLETE runs, if possible
     <<<
  <<
```

See Also
>>> STEADY-STATE

Command
```
>>> INDEX
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
Prints the iTOUGH2 command index to the iTOUGH2 output file.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print command INDEX
     <<<
  <<
```

See Also
```
LIST, HELP
```

Command

```
>>>> INDEX: index (index_i...)
```

or

```
>>>> PARAMETER: index (index_i...)
```

Parent Command

most third-level commands in block `> PARAMETER`

Subcommand

-

Description

This command provides a list of integers for further parameter specification. The integers are usually indexes of TOUGH2 arrays, such as *IPAR* in arrays *CP*(*IPAR*,*NMAT*) or *RPD*(*IPAR*), selecting the *IPAR*-th parameter of the capillary pressure or default relative permeability function, respectively. If multiple indexes are provided, a single parameter will be estimated and assigned to all the corresponding array elements.

Example

```
> PARAMETER
  >> estimate 2nd parameter of default CAPILLARY
     pressure function
     >>> DEFAULT
         >>>> PARAMETER CPD: 2
         <<<<
     <<<

  >> optimize generation RATE of "huff & puff" system
     >>> SOURCE: WEL_1
         >>>> ANNOTATION        : Injection
         >>>> INDEX of array F2 : 1 3 5 7 9
         <<<<
     >>> SOURCE: WEL_1
         >>>> ANNOTATION        : Pumping
         >>>> INDEX of array F2 : 2 4 6 8 10
         <<<<
     <<<
  <<
```

See Also

-

Command
`>>>> INDEX: index (index_i...)`
or
`>>>> PARAMETER: index (index_i...)`

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
This command provides a list of integers for further specification of user-specified observations (see command `>> USER (o)`).

Example
(see command `>> USER (o)`)

See Also
`>> USER(o)`

Command
>> INITIAL (PRESSURE/: *ipv*)

Parent Command
> PARAMETER

Subcommand
>>> DEFAULT
>>> MATERIAL

Description
This command selects as a parameter the initial condition for all grid blocks associated with a certain rock type (TOUGH2 variable *DEPU(ipv)*) or default initial condition (TOUGH2 variable *DEP(ipv)*). Since boundary conditions are specified as initial conditions for inactive grid blocks or grid blocks with a large volume, this command can also be used to estimate boundary conditions. Initial conditions estimates cannot be provided for individual elements unless they have a unique material name associated with them. Estimating the first primary variable can be selected using keyword PRESSURE. All the other primary variables must be identified by number, i.e. by an integer *ipv* that follows a colon on the command line. Alternatively, *ipv* can be provided through command >>>> INDEX. The initial guess for the parameter is taken from TOUGH2 block PARAM.4, or should be provided by using commands >> GUESS or >>>> GUESS or >>>> PRIOR.

Example
```
> PARAMETER
  >> INITIAL PRESSURE
     >>> MATERIAL: BOUND
         >>>> ANNOTATION: Boundary Pressure
         >>>> scale pressures on boundary by constant FACTOR
         <<<<
     <<<

  >> INITIAL condition for primary variable No.: 2
     >>> MATERIAL             : SAND1 DEFAU
         >>>> ANNOTATION      : Initial Saturation
         >>>> VALUE
         >>>> initial GUESS   : 10.4
         >>>> admissible RANGE: 10.01 10.99
         <<<<
     <<<
  <<
```

See Also
-

Command
>>> INPUT

Parent Command
>> CONVERGE

Subcommand
-

Description
This command makes iTOUGH2 stop immediately after the TOUGH2 and iTOUGH2 input files have been read and checked for consistency. This is useful to check input before time consuming simulations are invoked, or in combination with LIST, HELP, and >>> INDEX.

Example
```
> COMPUTATION
  >> print LIST of available commands on this command level
  >> CONVERGE (print HELP message to iTOUGH2 output file and
     >>> stop immediately after INPUT is read)
     <<<
  <<
```

See Also
>>> INDEX, HELP, LIST

Command
`>>> INTERFACE: elem_1 elem_2 (elem_i elem_j ...) (+ iplus)`

Parent Command
`>> FLOW`

Subcommand
any fourth-level command in block `> OBSERVATION`

Description
(synonym for command `>>> CONNECTION`)

Example
(see command `>>> CONNECTION`)

See Also
`>> CONNECTION`

Command
`>>>> INSTRUCTION: num-instruction-files (NO DELETE)`

Parent Command
`>>> PEST (c)`

Subcommand
-

Description
iTOUGH2 capabilities can be applied to non-TOUGH2 models using the PEST interface [*Doherty*, 2002]. Instruction files are used to communicate between the output variables of the user-supplied model (which must be provided through one or multiple ASCII text files) and the iTOUGH2 observation vector. (Note that if the model writes output to the screen, the '>' symbol can be used on the command line (see `>>>> EXECUTABLE`) to redirect standard output from the screen to a text file.)

An instruction file must be provided for each output file that contains an observable variable used by iTOUGH2 for model evaluation; *`num-instruction-files`* is the number of instruction files provided. Instruction files are matched to their corresponding output files on the lines following the `>>>> INSTRUCTION` command.

Upon initialization, all output files are deleted unless keyword `NO DELETE` is present.

PEST requires that at least one PEST observation is provided. However, if the external model is a preprocessor to TOUGH2, all observations may be internally taken from TOUGH2 arrays, without the need for an observation that is read from an external output file. Dummy instruction and output files can be automatically generated by providing their respective file names as *dummy.ins* and dummy.out. A "measured" value of zero will be generated and returned for comparison to an observation named `dummy` that needs to be defined in the iTOUGH2 input file.

Under Unix, it is recommended to add separate lines with the keyword `FILE:` followed by the file name, so the files are automatically copied to the temporary directory. This can occur anywhere in the iTOUGH2 input file.

Example

```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE  : "idratherusetough.exe > useless"
         >>>> TEMPLATE    : 1
              input.tpl     input.txt

         >>>> INSTRUCTION : 2
              output1.ins   useless
              output2.ins   some_results.txt
         <<<<

copy FILE: input.tpl            to temporary directory
copy FILE: output1.ins          to temporary directory
copy FILE: output2.ins          to temporary directory
copy FILE: idratherusetough.exe to temporary directory
```

The following example shows the use of dummy instruction and output files:

```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE  : TOUGH-pre-processor
         >>>> TEMPLATE    : 1
              meshgen.tpl   meshgen.txt

         >>>> INSTRUCTION : 1   NO DELETE
              dummy.ins     dummy.out
         <<<<
```

A dummy PEST observation needs to be provided as follows:

```
> OBSERVATION
  >> PEST
     >>> UNIVERSAL
         >>>> DATA
              dummy 0.0 1.0E-20
         <<<<
```

See Also

>>>> EXECUTABLE, >>>> TEMPLATE

Command
`>>> ITERATION: max_iter`

Parent Command
`>> CONVERGE`

Subcommand
-

Description
This command sets the maximum number of iTOUGH2 iterations to *max_iter*. If using the Levenberg-Marquardt minimization algorithm, an iTOUGH2 iteration consists of a number of TOUGH2 simulations and includes the following steps:

(1) solution of the forward problem,

(2) evaluation of the Jacobian matrix (requiring n or $2n$ TOUGH2 simulations depending on whether a forward or centered finite difference quotient is requested, see command `>> JACOBIAN`),

(3) updating of the parameter vector (see also command `>>> STEP`),

(4) check run(s) to see whether the new parameter set leads to a reduction of the objective function; if not, go back to step 3.

If the objective function is successfully reduced, the iteration is completed, and the last check run is used as the solution of the forward problem (step (1) above) for the next iteration. By default, new iterations are performed until one of the following convergence criteria is met (note that different convergence criteria apply if options other than Levenberg-Marquardt optimization are used):

(1) the maximum number of TOUGH2 simulations is reached (see command `>>> SIMULATION`),

(2) the maximum number of incomplete TOUGH2 simulations is reached (see command `>>> INCOMPLETE`),

(3) the scaled step size is smaller than the minimum relative step size = 10^{-9},

(4) all parameters are at their user-specified bounds,

(5) the objective function is smaller than the relative function tolerance,

(6) the maximum number of unsuccessful uphill steps is exceeded (see command `>>> UPHILL`),

(7) the Levenberg parameter exceeds 10^{12},

(8) the norm of the gradient vector is smaller than 10^{-5} (optimality criterion).

In most cases, however, it is sufficient to stop the inversion after a few iterations because no significant fit improvement is obtained after about 5 to 15 iterations. Generally more iterations are required with increasing number of parameters and stronger non-linearities of the flow problem.
The progress of the objective function reduction can be observed by typing the command `prista` (see unix script file *prista*), and inversion can be terminated using the `kit` command (see unix script file *kit*).
It is suggested to perform a single iTOUGH2 iteration or to use option `>> SENSITIVITY ANALYSIS` prior to running a full inversion in order to check the relative importance and sensitivity of each parameter, the parameter step size, the initial value of the Levenberg parameter, etc.

Example

```
> COMPUTATION
  >> STOP after
     >>> : 6 ITERATIONS
     <<<
  <<
```

See Also

`>>> SENSITIVITY, >>> CENTERED, >>> FORWARD, >>> INCOMPLETE, >>> SIMULATION, >>> UPHILL`

Command
>>>> ITERATION: *max_iter*

Parent Command
>>> ANNEAL

Subcommand
-

Description
This command limits the maximum number of iterations performed by the Simulated Annealing minimization algorithm. An iteration is completed if:

(1) the maximum number of steps m_{step} on a temperature level is reached (see command >>>> STEP (a)), or

(2) the objective function has been reduced $0.2 \cdot m_{step}$ times

Each iteration is followed by a reduction of the control parameter τ (temperature) according to the annealing schedule (see command >>>> SCHEDULE).

Example
```
> COMPUTATION
  >> OPTION
     >>> Simulated ANNEALing
         >>>> maximum number of ITERATIONS:  100
         >>>> maximum number of STEPS: 50
         >>>> annealing SCHEDULE:   0.95
         >>>> initial TEMPERATURE: -0.02
         <<<<
     <<<
  <<
```

See Also
>>>> STEP (a), >>>> SCHEDULE

Command
>>>> ITERATION: *niter*

Parent Command
>>> SELECT

Subcommand
-

Description
This command defines the number of iterations after which the criteria for automatic parameter selection is reevaluated (see commands >>>> CORRELATION and >>>> SENSITIVITY).
A full Jacobian matrix is evaluated every multiple of *niter* iterations. In intermediate iterations, only the columns of the Jacobian corresponding to the selected parameters are updated.

Example
```
> COMPUTATION
  >> OPTION
     >>> SELECT parameter automatically every
         >>>> : 3 ITERATIONS
         >>>> using SENSITIVITY criterion with rsens : -0.10
         <<<<
      <<<
  <<
```

See Also
>>>> CORRELATION, >>>> SENSITIVITY

Command
```
>> JACOBIAN
```

Parent Command
```
> COMPUTATION
```

Subcommand
```
>>> CENTERED
>>> FORWARD
>>> HESSIAN
>>> LIST
>>> PERTURB
```

Description
This is the parent command of a number of subcommands that deal with the calculation of the Jacobian matrix **J** . The elements of the Jacobian matrix are the partial derivatives of the system response at the calibration points with respect to the parameters to be estimated:

$$J_{ij} = -\frac{\partial r_i}{\partial p_j} = -\frac{\partial (z_i * - z_i)}{\partial p_j} = \frac{\partial z_i}{\partial p_j}$$

The Jacobian matrix discussed here must be distinguished from the one calculated in the simulation program TOUGH2. The latter is used to solve the set of non-linear algebraic iterations arising at each time step; its elements are the partial derivatives of the mass residuals with respect to the primary variables, and its numerical computation is controlled by the TOUGH2 variable *DFAC*.

Example
```
> COMPUTATION...
  >> ...of the JACOBIAN matrix is performed using...
     >>> FORWARD finite difference quotients for: 3
         iterations before switching to centered finite
         difference quotients.
     >>> Parameter PERTURBation factor is: 1 % (default)
     <<<
  <<
```

See Also
-

Command
```
>>> JACOBIAN
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints the Jacobian matrix after each iteration. By default, the (scaled) Jacobian matrix is printed only once at the end of the optimization.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print JACOBIan after each iteration.
     <<<
  <<
```

See Also
```
>>> RESIDUAL
```

Command
```
>> KLINKENBERG
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the Klinkenberg slip factor (TOUGH2 variable *GK(NMAT)*).

Example
```
> PARAMETER
  >> KLINKENBERG slip factor
     >>> ROCK type      : GRANI
         >>>> LOGARITHM
         >>>> RANGE     : 1.0 10.0
         >>>> VARIATION : 1.0
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>> L1-ESTIMATOR
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects the L_1-estimator, i.e., the objective function to be minimized is the sum of the weighted absolute residuals.

$$S = \sum_{i=1}^{m} \frac{|r_i|}{\sigma_i}$$

Minimizing the mean absolute deviation leads to a maximum-likelihood estimate if the errors follow a double exponential distribution.

$$\varphi(r_i) = \frac{1}{2\sigma_i} \exp\left(-\left| \frac{r_i}{\sigma_i} \right| \right)$$

The L1-estimator should be used, for example, to minimize a cost function for the optimization of a cleanup operation. Furthermore, it can be used whenever the objective function of interest is a linear function of the model output (e.g., in a sensitivity analysis using command `>>> OBJECTIVE` in block `>> OPTION`).
Note that this objective function is usually minimized using the Levenberg-Marquardt algorithm, which is designed for a quadratic objective function. Minimization is therefore rather inefficient, requiring more iterations and a high initial Levenberg parameter.

Example
```
> COMPUTATION
  >> OPTION
     >>> use L1-ESTIMATOR, then draw contours of the
     >>> OBJECTIVE function based on: 10 points
         along each axis of the parameter space
     <<<
  <<
```

See Also
```
>>> ANDREWS, >>> CAUCHY, >>> LEAST-SQUARES,
>>> QUADRATIC-LINEAR
```

Command
```
>> LAG
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
>>> SET
```

Description
This command selects as a parameter a constant lag, shifting data in time. The time lag is applied to the output that refers to a specific data set:

$$z(t) = z_{TOUGH2}(t + lag)$$

Here, t is time, lag is the estimated time lag in seconds, and z_{TOUGH2} is the TOUGH2 output. The result z is compared to the measurement z^* of the corresponding data set.
A single data set is identified by number using command `>>> SET (p)`; multiple data sets are specified with command `>>>> INDEX (p)`.
If the lag is known, i.e., does not need to be estimated, use command `>>>> SHIFT TIME`, a subcommand of `> OBSERVATION`.

Example
```
> PARAMETER
  >> LAG
     >>> NONE
         >>>> ANNOTATION: time lag
         >>>> INDEX: 1 2 3
         >>>> GUESS: 300 sec.
         <<<<
     <<<
  <<
```

See Also
```
>> DRIFT, >> SCALING, >> SHIFT, >>> SET (p), >>>> INDEX(p),
>>>> SHIFT
```

Command

```
>>> LATIN HYPERCUBE SAMPLING (MATRIX: ndim (iTOUGH2))
    (CORRELATION/COVARIANCE)
```

Parent Command

```
>> ERROR
```

Subcommand

-

Description

This command must be used in combination with command `>>> MONTE CARLO` to invoke stochastic simulations of correlated or uncorrelated parameters using Latin Hypercube Sampling (LHS). LHS is a variant of Monte Carlo simulations to quantify the uncertainty of model predictions as a result of parameter uncertainty. Many parameter sets are generated, following a predefined covariance matrix $\mathbf{C}_{pp}$ describing uncertainty of selected input parameters. LHS is an efficient sampling strategy where the probability density functions of the input parameters are subdivided into sections of equal probability $1/n_{MC}$, with n_{MC} being the number of Monte Carlo simulations. Use command `>>>> RANGE` to defined the lower and upper cutoff of the parameter's probability density function. The sampled parameter values are then randomly combined to yield parameter sets that follow the predefined correlation structure. Details can be found in *Zhang and Pinder* [2003].
If none of the keywords `COVARIANCE`, `CORRELATION`, or `MATRIX` is present, LHS uses the standard deviations provided in block `> PARAMETER` and assumes that the input parameters are uncorrelated. If parameter correlations are to be considered, matrix $\mathbf{C}_{pp}$ must supplied using one of the following options:

(1) Provide indices and elements of $\mathbf{C}_{pp}$. Example:

```
>>> LATIN HYPERCUBE SAMPLING, honor COVARIANCES
    1  1  0.80643E-04
    2  2  0.71921E-04
    2  1  0.64412E-04
```

Use keyword `CORRELATION` if off-diagonal term is correlation coefficient instead of covariance. Example:

```
>>> LATIN HYPERCUBE SAMPLING, honor CORRELATION coefficent
    1  1  0.80643E-04
    2  2  0.71921E-04
    2  1  0.864
```

(2) Provide keyword `MATRIX`, followed by a colon and the dimension *ndim* of the square matrix $\mathbf{C}_{pp}$. The lower triangle of the covariance matrix is then provided on exactly *ndim* additional lines. If keyword `CORRELATION` is present, the off-diagonal terms represent correlation coefficients rather than covariances. Example:

```
>>> LATIN HYPERCUBE, dimension of CORRELATION MATRIX: 4
    9.1234
   -0.67     0.00413
    0.80     0.213     1.3E-6
    0.50    -0.155     0.90     4.3E12
```

(3) If calculated during a previous iTOUGH2 inversion, the covariance matrix can be taken from the iTOUGH2 output file and directly copied after the command line. This option is invoked by keyword `iTOUGH2`. The matrix will be read by formatted input, so it is crucial that the correct format is maintained. If *ndim* is greater than 6, the matrix is split in multiple submatrices. All submatrices must be copied exactly as they were printed to the iTOUGH2 output file. Example:

```
>>> LATIN HYPERCUBE, MATRIX of dim.: 3 in iTOUGH2 format
                log(abs. perm.)    POROSITY SAND    Gas entrapped
log(abs. perm.)        .80643E-04           .846           -.253
POROSITY SAND          .64412E-04       .71921E-04         -.500
Gas entrapped         -.52623E-05      -.98296E-05       .53843E-05
```

Example

```
> COMPUTATION
  >> STOP
     >>> number of Monte Carlo SIMULATIONS: 20
     <<<
  >> ERROR propagation analysis
     >>> MONTE CARLO simulations, SEED number: 777
     >>> LATIN HYPERCUBE, dimension of CORRELATION MATRIX: 3
         9.1234
        -0.67     0.00413
         0.80     0.213     1.3E-6
     <<<
  <<
```

See Also

>>> FOSM, >>> EMPIRICAL ORTHOGONAL FUNCTIONS, >>> MONTE CARLO

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

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects least-squares optimization, i.e., the objective function to be minimized is the sum of the squared weighted residuals.

$$S = \mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r} = \sum_{i=1}^{m} \left(\frac{r_i}{\sigma_i} \right)^2$$

Minimizing the squared weighted residuals leads to a maximum-likelihood estimate if the errors are normally distributed with zero mean and covariance matrix $\mathbf{C}_{zz}$:

$$\varphi(\mathbf{r}) = (2\pi)^{-m/2} \left| \mathbf{C}_{zz} \right|^{-1/2} \exp\left(-\frac{1}{2} \mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r} \right)$$

Least-squares estimation is the default. If outliers are more prominent than described by the tail of the normal distribution, one may want to use one of the robust estimators to reduce the weight of outliers or even eliminate them (see commands `>>> ANDREWS`, `>>> CAUCHY`, or `>>> QUADRATIC-LINEAR`).
The least-squares objective function is a quadratic function if the residuals depend linearly on the parameters; it is nearly-quadratic for a non-linear model. The Levenberg-Marquardt algorithm is best suited to minimize the non-linear least-squares objective function.

Example
```
> COMPUTATION
  >> OPTION
     >>> use LEAST-SQUARES
     <<<
  <<
```

See Also
```
>>> ANDREWS, >>> CAUCHY, >>> L1-ESTIMATOR,
>>> QUADRATIC-LINEAR
```

Command
```
>>> LEVENBERG: lambda
```

Parent Command
```
>> CONVERGE
```

Subcommand
-

Description
This command sets the initial value of the Levenberg parameter λ (default: 1.0). During the optimization process (see command `>>> LEVENBERG-MARQUARDT`), the Levenberg parameter is divided by the Marquardt parameter ν (see command `>>> MARQUARDT`) after each successful iteration, and is multiplied by ν if the new parameter set leads to an increased value of the objective function, i.e., if an unsuccessful step was proposed.
A big value for λ means that a small step along the steepest descent direction is performed. A λ value of zero is equivalent to a Gauss-Newton step. The former is robust but inefficient; the latter has a quadratic convergence rate but may lead to unsuccessful steps.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> maximum number of ITERATIONS: 10
     >>> set initial LEVENBERG parameter to: 100.0 to make a
         safe first step
     <<<
   <<
```

See Also
```
>>> MARQUARDT
```

Command

```
>>> LEVENBERG-MARQUARDT (IDENTITY/EIGENVALUE)
    (SUPER/TRUNCATE (: (-)trunc))
```

Parent Command

```
>> OPTION
```

Subcommand

-

Description

This command selects the Levenberg-Marquardt algorithm to minimize the objective function. This is the default minimization algorithm. The Levenberg-Marquardt algorithm combines the robustness of a steepest descent method with the efficiency of a Gauss-Newton step (see command `>>> GAUSS-NEWTON`):

$$\Delta\mathbf{p} = \left(\mathbf{J}^T\mathbf{C}_{zz}^{-1}\mathbf{J} + \lambda\mathbf{D}\right)^{-1}\mathbf{J}^T\mathbf{C}_{zz}^{-1}\mathbf{r}$$
$$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \Delta\mathbf{p}$$

where $\mathbf{D}$ is the Tikhonov matrix. The Tikhonov matrix is either a diagonal matrix with the elements of the Fisher Information matrix $(\mathbf{J}^T\mathbf{C}_{zz}^{-1}\mathbf{J})$ (default), the identity matrix (keyword `IDENTITY`), or a diagonal matrix with the inverse of the squared eigenvalues of the Fisher Information matrix, scaled by the maximum squared eigenvalue (keyword `EIGEN`). The parameter space can be truncated (keyword `TRUNCATE`) based on the eigenvalues of the Fisher Information matrix. Parameter *trunc* defines the cut-off value as a fraction of the eigenvalue to the maximum eigenvalue (default: 10^{-6}). If given as an integer, the top INT(*trunc*) parameters will be selected. If *trunc* is a negative integer, more parameters are added as iterations proceed, with all parameters included for the final iteration. If keyword `SUPER` is present, superparameters will be created (see *Tonkin and Doherty* [2005]).

The Levenberg-Marquardt method switches continuously from a gradient method (large λ, see command `>>> LEVENBERG`) far from the minimum to a Gauss-Newton step as the minimum is approached and λ is reduced.

Example

```
> COMPUTATION
  >> OPTION
     >>> use LEVENBERG-MARQUARDT algorithm (default)
     <<<
```

See Also

```
>>> ANNEAL, >>> GAUSS-NEWTON, >>> GRID SEARCH, >>> SIMPLEX,
>>> LEVENBERG, >>> MARQUARDT, >>> HARMONY-SEARCH,
>>> EVOLUTIONARY
```

Command
```
>>> LINEARITY (: alpha (%))
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
The covariance matrix of the estimated parameters, $\mathbf{C}_{pp}$, is calculated using linear error analysis:

$$\mathbf{C}_{pp} = s_0^2 (\mathbf{J}^T \mathbf{C}_{zz}^{-1} \mathbf{J})^{-1}$$

The confidence region around the estimated parameter set for the linearized case consists of those values $\mathbf{p}$ for which

$$(\mathbf{p} - \hat{\mathbf{p}})^T \mathbf{C}_{pp} (\mathbf{p} - \hat{\mathbf{p}}) \le n \cdot F_{n,m-n,1-\alpha}$$

where $\hat{\mathbf{p}}$ is the parameter vector at the optimum. The covariance matrix $\mathbf{C}_{pp}$ approximates the actual surface of the objective function at its minimum by a tangent hyperellipsoid under the assumption of normality and linearity. If the model is nonlinear, the coverage of the confidence region by the linear approximation may be very poor with respect to both its size and its shape.

When using this command, it is assumed that the shape of the confidence region is close to ellipsoidal, and that the orientation of the hyperellipsoid in the *n*-dimensional parameter space is accurately obtained from the linear error analysis. Then, by only adjusting the size of the hyperellipsoid, we can better approximate the confidence region without losing the advantage of producing easily understandable results, which are also simple to report. The procedure adopted here is based on a comparison of the actual objective function with the results from the linear approximation at discrete points in the parameter space. These test points $\tilde{\mathbf{p}}$ are located along the main axis of the hyperellipsoid, i.e.:

$$\tilde{\mathbf{p}}_{i\pm} = \hat{\mathbf{p}} \pm (n \cdot F_{n,m-n,1-\alpha})^{1/2} a_i \mathbf{u}_i \, i = 1, \cdots, n$$

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 $\mathbf{C}_{pp}$. 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 determine the length of the semiaxis, are now corrected as follows:

$$a'^2_i = a_i^2 s_0^2 \left(\frac{A_+ + A_-}{2} \right)_i$$

with

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

Finally, the new covariance matrix is back-calculated from the eigenvectors $\mathbf{u}_i$ and the updated eigenvalues a'^2_i.

This correction procedure 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 $S(\tilde{\mathbf{p}}_{i+})$ and $S(\tilde{\mathbf{p}}_{i-})$ provide, as a byproduct of the correction procedure, some insight into the asymmetry of the true confidence region.

The user may specify α or $1-\alpha$ in % or the quantile $(n \cdot F_{n,m-n,1-\alpha})^{1/2}$ directly (by omitting `%` on the command line).

Note that the correction procedure fails if the minimum is not accurately identified. In this case iTOUGH2 automatically proceeds with the covariance matrix from the linear error analysis.

Example

```
> COMPUTATION
  >> ERROR analysis
     >>> calculate finite difference HESSIAN and then
     >>> check LINEARITY assumption on: 95 % confidence level
     <<<
  <<
```

See Also

```
>>> HESSIAN
```

Command
`LIST` (available on all command levels)

Parent Command
-

Subcommand
-

Description
This command provides a list of all available commands on the corresponding command level. The list contains the actual upper case spelling of the command as interpreted by the program. Potential keywords are not listed. A complete list of all commands can be obtained with command `>>> command INDEX`.

Example
```
> LIST
> COMPUTATION
  >> STOP after
     >>> INPUT
     <<<
  <<
<
```

The following list is printed to the iTOUGH2 output file showing any first-level command:

```
*****  LIST OF COMMANDS  *****   on level 1:
*     <
*     > OPTION
*     > COMPUTATION
*     > PARAMETER
*     > OBSERVATION
*     > MEASUREMENT
*
*****  LIST OF COMMANDS  *****
```

See Also
`>>> INDEX, >>> INPUT`

Command
`>>>> LOGARITHM`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
The parameter p to be estimated is the logarithm of the TOUGH2 parameter X:

$$p = \log_{10}(X) \Leftrightarrow X = 10^{p}$$

Estimation of logarithms is recommended if the parameter is expected to vary over a large range of values, suggesting a log-normal distribution of its estimate. Note that all quantities referring to this parameter are also in log-space (e.g., range, standard deviation, step length, etc.). Estimating the logarithm is chosen her eover estimating the parameter value directly (command `>>>> VALUE`), estimating a multiplication factor (command `>>>> FACTOR (p)`), or estimating a log-normally distributed multiplication factor (command `>>>> LOG(F)`.

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: SAND1
         >>>> estimate LOGARITHM
         >>>> initial GUESS      : -12.0 = 1 darcy
         >>>> admissible RANGE   : -15.0  -9.0 in log space
         >>>> standard DEVIATION :   1.0 order of magnitude
         >>>> maximum STEP size  :   0.5 log-cycles
         <<<<
     <<<
  <<
```

See Also
`>>>> FACTOR (p), >>>> LOG(F), >>>> VALUE`

Command
>>>> LOGARITHM

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command takes the natural logarithm of an observable variable. The corresponding measurement error is assumed to be log-normally distributed. Taking the logarithm is suggested when the observation assumes values over many orders of magnitude (e.g., concentration, water potential). Note that this option emphasizes the importance of smaller values of the variable, and is not applicable to data sets that contain zero measurements. Taking the natural logarithm is a special case of the Box-Cox transformation with parameter λ set to zero.

Example
```
> OBSERVATION
  >> CAPILLARY PRESSURE
     >>> ELEMENT: TP__1
         >>>> first take the ABSOLUTE value and
         >>>> then the LOGARITHM
         >>>> standard DEVIATION of untransformed data: 1.0E4
         >>>> DATA on FILE: wp.dat
         <<<<
     <<<
  <<
```

See Also
-

Command
`>>>> LOG(F)`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
The parameter to be estimated is a log-normally distributed factor with which the initial TOUGH2 parameter value is multiplied:

$$p = \log_{10}\left(X/X_0\right) \Leftrightarrow X = 10^p \cdot X_0$$

Here, p is the estimated parameter, X is the TOUGH2 parameter, and X_0 is the initial value of the TOUGH2 parameter. This option is useful to determine the mean of a log-normally distributed quantity while maintaining ratios (e.g., if estimating a common factor applied to all three permeability values in a model domain, the anisotropy ratio remains constant). Estimating a factor is chosen here over estimating the parameter value directly (command `>>>> VALUE`) or its logarithm (command `>>>> LOGARITHM (p)`).

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: SAND1 SAND2 SAND3 SAND4 SAND5
         >>>> estimate LOG(F)
         >>>> INDEX        : 1 2 3
         >>>> initial GUESS: 0.0  (default)
         >>>> RANGE        : -3.0  3.0
         <<<<
     <<<
  <<
```

See Also
`>>>> LOGARITHM (p), >>>> FACTOR, >>>> VALUE`

Command
```
>>> MARQUARDT: nue
```

Parent Command
```
>> CONVERGE
```

Subcommand
-

Description
This command sets the Marquardt parameter ν (default: 10.0). During the optimization process, the Levenberg parameter λ (see command `>>> LEVENBERG`) will be divided by the Marquardt parameter ν after each successful iteration, and it will be multiplied by ν if the new parameter set leads to an increased value of the objective function, i.e., if an unsuccessful step was proposed.
A big value for λ means that a small step along the gradient direction is performed. A λ value of zero is equivalent to a Gauss-Newton step. The former is robust but inefficient, the latter has a quadratic convergence rate but may lead to unsuccessful steps.
The Marquardt parameter therefore determines how fast the step size and step direction changes from steepest descent to Gauss-Newton and vice versa.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> maximum number of ITERATIONS: 10
     >>> set initial LEVENBERG parameter to: 0.1 to make a
         safe first step
     >>> MARQUARDT parameter : 2.0 (slow change to
         Gauss-Newton steps)
     <<<
   <<
```

See Also
```
>>> LEVENBERG
```

Command

```
>> MASS FRACTION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> ELEMENT
```

Description

This command selects as an observation type the mass fraction of component *icomp* in phase *iphase*. This observation type refers to one or more elements.
Component number *icomp* or component name *comp_name*, and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the iTOUGH2 header, and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively.

Example

```
> OBSERVATION
  >> MASS FRACTION of BRINE in LIQUID
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> MASS FRACTION of COMPONENT No.: 2 in PHASE No.: 2
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> MASS FRACTION
     >>> ELEMENT: A1__1
         >>>> COMPONENT: 2
         >>>> PHASE    : 2
```

See Also

`>> MOLE FRACTION`, `>> CONCENTRATION`

Command
>>> MATERIAL: *mat_name* (*mat_name_i ...*) (+ *iplus*)
or
>>> ROCKS : *mat_name* (*mat_name_i ...*) (+ *iplus*)

Parent Command
any second-level command in block > PARAMETER referring to a material name

Subcommand
any fourth-level command in block > PARAMETER

Description
This command identifies material names (TOUGH2 variable *MAT*). Most parameters refer to a particular material, i.e., they are specified in TOUGH2 in block ROCKS. Rock types are designated by a five-character code name. Blanks in the material name must be replaced by underscores (e.g., if *MAT* in TOUGH2 reads 'CLAY ', *mat_name* must read 'CLAY_'). If multiple material names are provided, the estimate of the corresponding parameter will be jointly assigned to all listed materials. If distinct parameters are sought for each rock type, separate >>> MATERIAL blocks must be defined. Default properties (i.e., for parameters referring to TOUGH2 blocks PARAM.4 and RPCAP) can be selected either by >>> DEFAULT or >>> MATERIAL: DEFAU. If the last two characters of the last *mat_name* are integers, a sequence of *iplus* material names can be generated. The following two command lines are thus identical:

```
>>> MATERIAL: BOREH ROC10 ROC_1 +4
>>> MATERIAL: BOREH ROC_1 ROC_2 ROC_3 ROC_4 ROC_5 ROC10
```

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL: BOUND BOREH SAN_1 +2
         >>>> LOGARITHM
         <<<<
     <<<
  >> INITIAL PRESSURE
     >>> MATERIAL: BOREH
         >>>> ANNOTATION: Pi borehole
         <<<<
     >>> MATERIAL: BOUND DEFAU
         >>>> ANNOTATION: Pi elsewhere
         <<<<
```

See Also
>>> DEFAULT, >>> MODEL

Command
`>>>> MEAN (VOLUME)`

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
(synonym for command `>>>> AVERAGE`)

Example
(see command `>>>> AVERAGE`)

See Also
`>>>> AVERAGE`

Command
```
>> MINC
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects as a parameter the fracture spacing which is a parameter of the MINC preprocessor (TOUGH2 variable *PARMINC(I)*). The fracture set is identified through command `>>>> INDEX`. This parameter refers to the entire model.
Estimating fracture spacing is only possible if the mesh can be generated internally without further editing by the user. Single elements and connections can be modified, however, by means of TOUGH2 commands ELEM2 and CONN2, respectively. For example, if a mesh is generated using MESHMAKER and MINC, the volumes of individual boundary elements can be adjusted in the ELEM2 block, i.e., avoiding the need for external editing.

Example
```
> PARAMETER
  >> MINC parameter
     >>> MODEL
         >>>> ANNOTATION: Fracture spacing
         >>>> INDEX     : 1 2 3  (one value for all
                                   three fracture sets)
         >>>> GUESS     : 50.0 [m]
         >>>> DEVIATION : 10.0 [m]
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>> MODEL
```
or
```
>>> NONE
```

Parent Command
any second-level command in block `> PARAMETER` or `> OBSERVATION` addressing parameters or observations that refer to the entire model domain or a non-specific item.

Subcommand
any fourth-level command in block `> PARAMETER` or `> OBSERVATION`, respectively.

Description
Most parameters refer to a rock type or code name of a sink or source. Some parameters, however, are general or associated with the entire model domain. Similarly, while most observations refer to an element, connection, or sink/source, there are observation types that are not associated with a well-defined point in space. The commands `>>> MODEL` or `>>> NONE` are dummy commands, i.e. place holders on the third command level for parameter and observation types not further specified by one of the other third-level commands.

Example
```
> PARAMETER
  >> SELEC parameter
     >>> NONE
         >>>> ANNOTATION: Gel Density (EOS11)
         >>>> INDEX     : 2
         >>>> VALUE
         <<<<
     <<<
  <<

> OBSERVATION
  >> TOTAL MASS of PHASE: 2
     >>> refers to entire MODEL
         >>>> ANNOTATION   : Mass of liquid
         >>>> DATA on FILE : liquid.dat
         >>>> DEVIATION    : 0.01 [kg]
     <<<
   <<
```

See Also
`>>> ELEMENT`, `>>> CONNECTION`, `>>> DEFAULT`, `>>> MATERIAL`, `>>> SET`, `>>> SOURCE`

Command

```
>> MOLE FRACTION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase) (DISSOLVED)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> ELEMENT
```

Description

This command selects as an observation type the molre fraction of component *icomp* in phase *iphase*. (Note that brine is not treated as a component with a molecular weight; its molecular weight is arbitrarily assigned to that of water.) This observation type refers to one or more elements.

Component number *icomp* or component name *comp_name*, and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the iTOUGH2 header, and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively.

If keyword `DISSOLVED` is used, the mole fraction of the selected gaseous components in the liquid phase is returned without accounting for the water mole fraction.

Example

```
> OBSERVATION
  >> MOLE FRACTION of BRINE in LIQUID
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> MOLE FRACTION of COMPONENT No.: 2 in PHASE No.: 2
     >>> ELEMENT: A1__1
```

or

```
> OBSERVATION
  >> MOLE FRACTION
     >>> ELEMENT: A1__1
         >>>> COMPONENT: 2
         >>>> PHASE    : 2
```

See Also

```
>> CONCENTRATION, >> MASS FRACTION
```

Command

```
>> MOMENT (FIRST/SECOND) (X/Y/Z) (comp_name/COMPONENT: icomp)
          (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> MODEL
```

Description

This command selects as an observation type the first or second spatial moment in X, Y, or Z direction of component *icomp* or phase *iphase*. This observation type refers to all elements with 10,000 > *SPHT* > 0. The X, Y, and Z coordinates of the grid blocks must be given in TOUGH2 block ELEME, columns 51-60, 61-70, 71-80, respectively. Component number *icomp* or component name *comp_name*, and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the iTOUGH2 header, and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively. If only a phase but no component is specified, the spatial moment of the indicated phase including all components is calculated. If only a component but no phase is given, the spatial moment of that component in all phases is calculated. If both a component and a phase are given, the spatial moment of the component in the specified phase is calculated.

The spatial moments are calculated as follows:

$$M_{ijk} = \sum_{\beta} \sum_{\kappa} \sum_{n=1}^{NEL} \left(X^i \cdot Y^j \cdot Z^k \cdot V \cdot \phi \cdot S_\beta \cdot \rho_\beta \cdot X_\beta^\kappa \right)_n$$

where the first sum is taken over the selected phase(s) β, the second sum is taken over the selected component(s) κ, and the third summation accumulates the masses of component κ in phase β from all elements that are included in the global mass balance calculation (i.e., elements with a rock grain specific heat lower than 10^4 J/kg ºC). The mass in element *n* is the product of the element volume V_n, the porosity ϕ_n, the saturation $S_{\beta n}$, the phase density $\rho_{\beta n}$, and the mass fraction $X_{\beta n}^\kappa$.

The first moment represents the center of mass coordinates, which are given by

$$\langle X \rangle = M_{100} / M_{000}$$
$$\langle Y \rangle = M_{010} / M_{000}$$
$$\langle Z \rangle = M_{001} / M_{000}$$

The second moment represents the variances in the three directions given by

$$\sigma_X^2 = \frac{M_{200}}{M_{000}} - \langle X \rangle^2$$
$$\sigma_Y^2 = \frac{M_{020}}{M_{000}} - \langle Y \rangle^2$$
$$\sigma_Z^2 = \frac{M_{002}}{M_{000}} - \langle Z \rangle^2$$

This option may be especially useful for characterizing the location and spreading of a contaminant or saturation plume. Note that boundary effects may have a strong impact on spatial moment calculation.

Example

```
> OBSERVATION
  >> FIRST MOMENT (Z-coord.) of TRACER
     >>> entire MODEL
         >>>> ANNOTATION:  plume Z coord.
         >>>> NO DATA
         <<<<
     <<<

  >> SECOND MOMENT (X-coord.) of TRACER
     >>> entire MODEL // (except elements with SPHT > 10000)
         >>>> ANNOTATION:  transversal spreading
         >>>> NO DATA
         <<<<
     <<<
  <<
```

See Also

-

Command
`>>> MONTE CARLO (SEED: iseed) (GENERATE) (CLASS: nclass)`

Parent Command
`>> ERROR`

Subcommand
-

Description
This command performs Monte Carlo simulations to determine the uncertainty of model predictions as a result of parameter uncertainty. The general procedure is as follows:

(1) a probability distribution is defined for each uncertain parameter;

(2) parameter values are randomly sampled from their respective distributions;

(3) sampled parameter values are randomly combined to obtain a parameter vector;

(4) a TOUGH2 simulation is performed and the results at the specified observation points are stored;

(5) steps (3) through (4) are repeated;

(6) the ensemble of the modeling results can be statistically analyzed.

The type of distribution can be specified for each parameter in the respective parameter definition block (see command `>>>> UNIFORM`, `>>>> NORMAL`, or `>>>> TRIANGULAR`). The mean is given by the initial guess (see command `>>>> GUESS`), and the standard deviation by command `>>>> DEVIATION (p)`. The generated parameter value is rejected if outside the specified range (see command `>>>> RANGE`).
The number of Monte Carlo simulations, n_{MC}, must be provided in block `>> CONVERGE` using command `>>> SIMULATION`, i.e., by specifying the maximum number of TOUGH2 simulations. The number of Monte Carlo simulations n_{MC} can be considered sufficient if:

(1) The selected probability density function of the input parameters is reasonably well approximated by the histogram of the randomly generated parameter values. If correlations among the parameters are to be included, use Empirical Orthogonal Functions (command `>>> EOF`). Keyword `GENERATE` can be used in combination with various seed numbers (keyword `SEED`) and values for n_{MC} (command `>>> SIMULATION`) to generate histograms of the input parameters without actually performing Monte Carlo simulations. Once a satisfactory distribution of the input parameters is achieved, the Monte Carlo simulations can be invoked by simply deleting keyword `GENERATE`.

(2) The histogram of the model predictions allows for a statistical analysis. That means that a sufficient number of realizations (= simulation results) should fall within each interval used to calculate probabilities. For example: The probability that TCE concentrations c_{TCE} fall within the interval $[a,b]$ is approximated by:

$$\Gamma_{[a,b]} = \Pr(a \le c_{TCE} \le b) = \frac{\text{number of realizations in interval } [a,b]}{\text{total number of Monte Carlo simulations}} = \frac{n_{[a,b]}}{n_{MC}}$$

Therefore, the minimum number of Monte Carlo simulations, $n_{MC,\min}$, should be large enough so that $\Gamma_{[a,b]}$ remains constant, i.e. independent of n_{MC}. This condition is fulfilled for relatively small values of n_{MC} in the case of intervals around the mean, where $n_{[a,b]}$ is usually large due to the high probability density. However, if we are interested in the tail of the distribution, for example to calculate the (low) risk that TCE concentrations exceed a certain standard, then the number of Monte Carlo simulations required is much higher.

(3) The minimum number of Monte Carlo simulations must be increased if the number of uncertain parameter increases because more parameter combinations are possible.

(4) From experience, the number of Monte Carlo simulations can be as low as 50 and as high as 2000 or greater.

Histograms of the input parameters and output variables as printed to the iTOUGH2 output file. By default, the interval between the smallest and highest value is subdivided into $\sqrt{n_{MC}}$ classes. The number of classes can be changed using keyword `CLASS`.

The standard plot file contains all output sets from n_{MC} simulations, where each set contains the model result as a function of time. A second plot file is generated with "*_mc*" in the file name. In this second plot file, each curve represents the ensemble of the model output at one point in time. Denoting the number of output sets, for which prediction uncertainty is to be studied, by n_{set}, and the number of times by n_{times}, then the standard plot contains $n_{set} \cdot n_{MC}$ curves, each curve consisting of n_{times} points. The second plotfile contains $n_{set} \cdot n_{times}$ curves, each consisting of n_{MC} points.

Example

```
> PARAMETER
  >> ABSOLUTE permeability
     >>> MATERIAL                : ROCK_
         >>>> generate NORMAL distribution
         >>>> of LOGARITHM
         >>>> with mean (GUESS)  : -16.0
         >>>> standard DEVIATION :   1.0
         >>>> in the RANGE       : -18.5  -13.5
         <<<<
      <<<
  >> POROSITY
     >>> MATERIAL                : ROCK_
         >>>> VALUE
         >>>> UNIFORM distribution
         >>>> RANGE               : 0.02 0.10
         <<<<
     <<<
  <<

> OBSERVATION
  >> TIMES: 20 EQUALLY spaced in MINUTES
     3.0 60.0

  >> PRESSURE
     >>> ELEMENT: PRE_0
         >>>> NO DATA
         >>>> WEIGHT: 1.0
         <<<<
     <<<
  <<

> COMPUTATION
>> ERROR propagation analysis by means of...
     >>> : 1000 MONTE CARLO (SEED number is: 7777) using...
     >>> LATIN HYPERCUBE sampling
     <<<
  <<
```

See Also

>>> EMPIRICAL ORTHOGONAL FUNCTIONS, >>> FOSM,
>>> LATIN HYPERCUBE, >>> SIMULATION, >>>> NORMAL, >>>> RANGE,
>>>> UNIFORM, >>>> TRIANGULAR

Command
```
>>> NEW OUTPUT
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command creates a new TOUGH2 output file with unique file name for each TOUGH2 run. By default, the TOUGH2 output file is overwritten each time a new simulation starts.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> write to NEW OUTPUT after each TOUGH2 run.
     <<<
  <<
```

See Also
-

Command
`>>> NONE`

Parent Command
any second-level command in block `> PARAMETER` or `> OBSERVATION` addressing parameters or observations that refer to the entire model domain or a non-specific item.

Subcommand
any fourth-level command in block `> PARAMETER` or `> OBSERVATION`, respectively.

Description
(synonym for command `>>> MODEL`)

Example
(see command `>>> MODEL`)

See Also
`>>> MODEL`

Command
`>>>> NORMAL`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
(synonym for command `>>>> GAUSSIAN`)

Example
(see command `>>>> GAUSSIAN`)

See Also
`>>>> GAUSSIAN, >>>> UNIFORM, >>>> TRIANGULAR`

Command
```
>>> OBJECTIVE
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints the value of the objective function after each TOUGH2 simulation. By default, the value of the objective function is printed to the output file only after completion of an iTOUGH2 iteration.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> always print OBJECTIVE function
     <<<
  <<
```

See Also
```
>>> JACOBIAN
```

Command
`>>> OBJECTIVE: ninval1 (ninvali...) (UNSORTED)`

Parent Command
`>> OPTION`

Subcommand
-

Description
(synonym for command `>>> GRID SEARCH`)

Example
(see command `>>> GRID SEARCH`)

See Also
`>>> GRID SEARCH`

Command

> OBSERVATION

Parent Command

-

Subcommands

```
>> CONCENTRATION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase)
>> CONTENT (phase_name/PHASE: iphase)
>> COVARIANCE (FILE: filename)
>> CUMULATIVE (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
>> DRAWDOWN (phase_name/PHASE: iphase)
>> ENTHALPY (phase_name/PHASE: iphase)
>> FLOW (phase_name/PHASE: iphase/HEAT)
>> GENERATION (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
>> MASS FRACTION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase)
>> MOLE FRACTION (comp_name/COMPONENT: icomp)
                 (phase_name/PHASE: iphase)
>> MOMENT (FIRST/SECOND) (X/Y/Z)
     (comp_name/COMPONENT:icomp) (phase_name/PHASE: iphase)
>> PEST
>> PRESSURE (CAPILLARY) (phase_name/PHASE: iphase)
>> PRODUCTION (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase)
>> REGULARIZATION (FILE: file_name) (BETA: beta)
>> RESTART TIME: ntime (time_unit) (NEW)
>> SATURATION (phase_name/PHASE: iphase)
>> SECONDARY (phase_name/PHASE: iphase) (: ipar)
>> TEMPERATURE
>> TIME: ntime (EQUAL/LOGARITHMIC) (time_unit)
>> TOTAL MASS (comp_name/COMPONENT: icomp)
              (phase_name/PHASE: iphase) (CHANGE)
>> USER (: anno)
>> VOLUME (phase_name/PHASE: iphase) (CHANGE)
>> WATERTABLE
```

Description

This is the first-level command for specifying the observations available for use in parameter estimation. It can also be used to specify the points in space and time at which output is requested for sensitivity analysis, uncertainty propagation analysis, or plotting. The observations are a subset of all TOUGH2 output variables. Only those variables should be

specified for which data are available. Besides the observations listed above, the user can specify additional data types by means of command `>> USER`.
The second-level command `>> TIME` is used to select calibration points in time. The observation type is specified by the second-level command, the points in space are identified by the third-level command, and further specifications as well as the data themselves are given through fourth-level commands. The generic structure of a observation block is as follows:

```
> OBSERVATION
  >> specify calibration points in TIME
  >> specify observation type
     >>> specify location
         >>>> provide details
         >>>> provide data
         <<<<
     <<<
  <<
```

Example
(see examples on second command level)

See Also
-

Command
```
>> OPTION
```

Parent Command
```
> COMPUTATION
```

Subcommand
```
>>> ANDREWS
>>> ANNEAL
>>> CAUCHY
>>> DIRECT
>>> FORWARD
>>> GAUSS-NEWTON
>>> GRID SEARCH
>>> L1-ESTIMATOR
>>> LEAST-SQUARE
>>> LEVENBERG-MARQUARDT
>>> LIST
>>> OBJECTIVE
>>> PARALLEL
>>> PEST
>>> PVM
>>> QUADRATIC-LINEAR
>>> SELECT
>>> SEP
>>> SENSITIVITY
>>> SIMPLEX
>>> STEADY-STATE
>>> SVD
>>> TORNADO
>>> WORTH
```

Description
This is the parent command of a number of subcommands for selecting iTOUGH2 program options. By default, iTOUGH2 performs automatic model calibration using the weighted least-squares objective function and the Levenberg-Marquardt minimization algorithm.

Example
(see examples on third command level)

See Also
-

Command
```
>> OUTPUT
```

Parent Command
```
> COMPUTATION
```

Subcommand
```
>>> BENCHMARK
>>> CHARACTERISTIC
>>> COVARIANCE
>>> FORMAT
>>> INDEX
>>> LIST
>>> JACOBIAN
>>> NEW OUTPUT
>>> OBJECTIVE
>>> PERFORMANCE
>>> PLOTFILE
>>> PLOTTING
>>> RESOLUTION
>>> SENSITIVITY
>>> time_unit
>>> UPDATE
>>> RESIDUAL
>>> VERSION
>>> WORTH
```

Description
This command specifies the format and amount of printout generated. The iTOUGH2 output file contains the results usually needed for subsequent interpretation. Additional information can be requested which may be useful for debugging.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> Generate plot files in : COLUMN FORMAT
     >>> plot CHARACTERISTIC curves
     >>> print OBJECTIVE function after each TOUGH2 run
     >>> print RESIDUALS after each iTOUGH2 iteration
     >>> print parameter IDENTIFIABILITY
     >>> perform data-WORTH analysis based on DETERMINANT
     <<<
```

See Also
-

Command
```
>> PARALLEL PLATE
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the aperture of a parallel plate fracture model. The single fracture must be modeled as a 1D- or 2D model of thickness 1 meter, i.e., aperture must be identical to porosity. Furthermore, the capillary pressure model must contain a parameter `CP(2,NMAT)` which represents the gas entry pressure (e.g., Brooks-Corey (`ICP`=10), van Genuchten (`ICP`=11)).
The parallel plate model relates porosity φ, permeability k [m^2], and gas entry pressure p_e [Pa] to aperture a [m] as follows:

$$\varphi = a$$

$$k = \frac{f \cdot a^3}{12}$$

$$p_e = \frac{2\sigma\cos(\phi)}{a} = \frac{0.14366}{a}$$

Here, $f = 1$ is fracture frequency [m^{-1}], $\sigma = 0.07183$ [Pa·m] is the surface tension of water, and ϕ is the contact angle which is assumed to be zero. Using this option overwrites the porosity, absolute permeability and parameter `CP(2,NMAT)` given in the TOUGH2 input file.
Note that the parallel plate model used here is inconsistent. Using a relative permeability and capillary pressure function presumes some surface roughness or aperture distribution within the fracture plane.

Example
```
> PARAMETER
  >> aperture in PARALLEL PLATE fracture model
     >>> ROCK type               :  FRACT
         >>>> LOGARITHM
         >>>> PRIOR information :  -4.0   (100 micron)
         >>>> standard DEVIATION:   1.0
         >>>> RANGE             :  -5.0 -3.0
         <<<<
     <<<
```

See Also
-

Command

```
>>> PARALLEL: ncores (LEVENBERG: ncoreslm / JACOBIAN)
      (SLEEP: isleep)
```

Parent Command

```
>> OPTION
```

Subcommand

-

Description

Allows embarrassingly parallel execution of forward simulations on multi-core machines. The number of cores ***ncores*** is provided on the command line. The parent process must not be included in the list. This option is similar to iTOUGH2-PVM [*Finsterle*, 1998], except that heterogeneous workstation clusters are not supported. PVM does not need to be installed.

Message passing occurs through temporary files written to and read from directory */tmp* (Unix/Linux) or *C:* (PC).

If the Levenberg-Marquardt algorithm is used, trial parameter sets obtained with different Levenberg parameters can be evaluated in parallel on *ncoreslm* processors using keyword LEVENBERG. By default, ***ncoreslm*** = min(6, ***ncores***). Keyword JACOBIAN instructs iTOUGH2 to only evaluate the Jacobian in parallel, i.e., ***ncoreslm*** = 1.

The parent process can be suspended for ***isleep*** seconds each time it checks for incoming residuals (default: ***isleep*** = 1).

Example

```
> COMPUTATION
  >> OPTION
     >>> use LEVENBERG-MARQUARDT minimization algorithm
     >>> PARALLEL: 5 (parallelize JACOBIAN only)
     <<<
  <<
```

See Also

```
>>> PVM
```

Finsterle [1998]

Command

> PARAMETER

Parent Command

-

Subcommands

>> ABSOLUTE PERMEABILITY
>> BIOT
>> BOTTOMHOLE PRESSURE
>> BOX-COX
>> BULK DENSITY
>> CAPACITY
>> CAPILLARY PRESSURE FUNCTION
>> COMPRESSIBILITY
>> CONDUCTIVITY (WET/DRY)
>> DIFFUSION
>> DILATION
>> DRIFT
>> ENTHALPY
>> EXTERNAL
>> FORCHHEIMER
>> FRICTION ANGLE
>> FRICTION CORRECTION FACTOR
>> FRICTION FACTOR
>> GEOT
>> GUESS (FILE: *file_name*)
>> HARDENING
>> IFS
>> INITIAL (PRESSURE/: *ipv*)
>> KLINKENBERG
>> KURTOSIS
>> LAG
>> LIST
>> MINC
>> PARALLEL PLATE
>> PARMULT
>> PARSHIFT
>> PEST
>> POISSON
>> POROSITY
>> PRODUCTIVITY INDEX
>> PUMPING RATIO
>> RATE
>> REGION (SINK/SOURCE, PERMEABILITY, OBSERVATION)

```
>> REGRESSION
>> REGULARIZATION (FILE: file_name) (BETA: beta)
>> RELATIVE PERMEABILITY FUNCTION
>> SCALE
>> SCALING
>> SELEC
>> SHEAR
>> SHIFT
>> SKEWNESS
>> SKIN
>> STRAIN
>> TIME
>> TORTUOSITY
>> USER (: anno)
>> VOID FRACTION
>> YIELD
>> YOUNG
```

Description
This is the first-level command to specify the parameters to be estimated. The parameters to be estimated are a subset of all TOUGH2 input parameters defined in the TOUGH2 input file. Only those parameters should be specified that are unknown or uncertain; they will be subjected to the estimation process or uncertainty propagation analysis. Select only those parameters that are sensitive enough to influence the state variables for which observations are available (see also `>>> SELECT`). Besides the parameters listed above, the user can specify additional parameters by means of command `>> USER`.
The parameter type is specified by the second-level command, the domain it refers to is identified by the third-level command, and further specifications must be provided through a number of fourth-level commands. The generic structure of a parameter block is as follows:

```
> PARAMETER
  >> specify parameter type
     >>> specify parameter domain
         >>>> provide details
         <<<<
     <<<
  <<
```

Example
(see examples for second-level commands)

See Also
`> OBSERVATION, > COMPUTATION`

Command
`>>>> PARAMETER: index (index_i...)`

Parent Command
any third-level command in block `> PARAMETER` and `> OBSERVATION`

Subcommand
-

Description
(synonym for command `>>>> INDEX`)

Example
(see command `>>>> INDEX`)

See Also
`>>>> INDEX`

Command
>>>> PARENT: *parent_ID*
or
>>>> RELATED to: *parent_ID*
or
>>>> TIED TO: *parent_ID*

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command ties the parameter to a parent parameter. The parameter gets its updated value from the parent parameter, which is identified by its sequence number *parent_ID* in the > PARAMETER block. The parameter itself is inactive. It is essential that the parent and daughter parameter have the same parameter transformation (e.g., both refer to the parameter value itself, its logarithm, or a multiplication factor).

Example
```
> PARAMETER
  >> RELATIVE permeability function (parent parameter)
     >>> MATERIAL: SAND
         >>>> ANNOTATION: Res. Liq. Sat
         >>>> INDEX: 1
         >>>> VALUE
         >>>> GUESS: 0.2
         <<<<
      <<<

  >> CAPILLARY pressure function (tied parameter)
     >>> MATERIAL: SAND
         >>>> ANNOTATION: Res. Liq. Sat
         >>>> INDEX: 1
         >>>> VALUE
         >>>> TIED TO par. No.: 1
         <<<<
      <<<

  <<
```

See Also
-

Command
```
>>> PERFORMANCE
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command performs a very rough benchmark analysis of computer performance and prints the relative CPU time requirement as compared to a reference workstation.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> perform PERFORMANCE comparison
     <<<
  <<
```

See Also
-

Command
`>>> PERTURB: (-)alpha(%)`

Parent Command
`>> JACOBIAN`

Subcommand
-

Description
This command specifies the perturbation factor α for numerical computation of the Jacobian matrix.

The columns of the Jacobian matrix are calculated by perturbing the corresponding parameter p_i by a small amount δp_i, and taking either a forward or centered finite difference quotient. The perturbation is usually a fraction of the parameter value itself:

$$\delta p_i = \alpha \cdot p_i$$

The default value for α is 1 %, which can be changed globally for all parameters using the third-level command `>>> PERTURB`, or individually for each parameter using the fourth-level command `>>>> PERTURB`. A negative value can be provided to specify a constant perturbation, independent of parameter value:

$$\delta p_i = |-\alpha|$$

This option is sometimes required for parameters having either very small or very large values (such as initial pressure, generation times, or residual gas saturation).

Example
```
> COMPUTATION
  >> JACOBIAN
     >>> PERTURBation factor alpha : 0.005 of parameter value
     <<<
  <<
```

See Also
`>>> FORWARD, >>> CENTERED, >>>> PERTURB`

Command
>>>> PERTURB: (-)*alpha*(%)

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command specifies the perturbation factor α for numerical computation of the Jacobian matrix. The columns of the Jacobian matrix are calculated by perturbing the corresponding parameter p_i by a small amount δp_i, and taking either a forward or centered finite difference quotient. The perturbation is usually a fraction of the parameter value itself:

$$\delta p_i = \alpha \cdot p_i$$

The default value for α is 1 %, which can be changed globally for all parameters using the third-level command >>> PERTURB, or individually for each parameter using the fourth-level command >>>> PERTURB. A negative value can be provided to specify a constant perturbation, independent of parameter value:

$$\delta p_i = |-\alpha|$$

This option is sometimes required for parameters having either very small or very large values (such as initial pressure, generation times, or residual gas saturation).

Example
```
> PARAMETER
  >> TIME at which production rate changes
     >>> SOURCE: WEL_3
         >>>> INDEX of array F1  :  5
         >>>> initial GUESS      :  1.7E8 seconds
         >>>> PERTURB by constant: -3.6E3 seconds
         <<<<
     <<<
  <<
> COMPUTATION
  >> JACOBIAN
     >>> all other parameters are PERTURBed by : 2 %
         of their respective values
     <<<
  <<
```

See Also
>>> PERTURB

Command
```
>> PEST
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
```

Description
This command signifies that a parameter related to a user-supplied model (i.e., not a TOUGH2 module) be selected. The parameter will be identified and updated using the template file of the PEST interface. An initial guess must be provided for all non-TOUGH2 parameters through commands `>>>> GUESS` or `>>>> PRIOR`. All PEST-related parameters must be specified before any TOUGH2-related parameters are selected.

Example
```
> PARAMETER
  >> PEST
     >>> NONE
         >>>> ANNOTATION: coefficient-A
         >>>> LOGARITHM
         >>>> GUESS     : -3.0
         >>>> RANGE     : -6.0 0.0
         <<<<
     <<<
  <<
```

See Also
`>> PEST (o)`, `>>> PEST`

Command
```
>> PEST
```

Parent Command
```
> OBSERVATION
```

Subcommand
```
>>> NONE
>>> MODEL
>>> UNIVERSAL
```

Description
This command selects an (unknown) observation type related to a user-supplied model. The calculated value is identified and extracted from the output files using the search directives of a PEST instruction file. Unique observation names must be provided in the first data column, followed by the measured values, and (optionally) the weight attached to the residual. The case-insensitive observation names must be identical to those used in the instruction file(s). All PEST-related observations must be specified before any TOUGH2-related observations are selected.

Example
```
> OBSERVATION
  >> PEST
     >>> UNIVERSAL
         >>>> ANNOTATION  :   Total Costs
         >>>> DATA
              capital-cost    0.0
              operating-cost  0.0
         >>>> WEIGHT      :   1.00531 [dollar/CHF]
         <<<<

     >>> UNIVERSAL: pumping rates
         >>>> DATA
              pH-after-0-yr   7.2  1.0  pump
              pH-after-1-yr   5.8  0.5  pump
              pH-after-2-yr   3.6  0.5  pump
         <<<<
      <<<
```

See Also
>> PEST (p), >>> PEST

Command

```
>>> PEST
```

Parent Command

```
>> OPTION
```

Subcommand

```
>>>> DECPOINT
>>>> EXECUTABLE
>>>> INSTRUCTION
>>>> PRECISION
>>>> TEMPLATE
```

Description

This command invokes fourth-level commands to specify files and options for the PEST/JUPITER interface between iTOUGH2, a user-supplied model, and its input and output files. It is used to (1) identify the executable (or script or batch) file that calls the forward model, (2) relate the template file(s) to the corresponding input file(s), (3) relate the instruction file(s) to the corresponding output file(s), and (4) to specify the precision with which parameter values are to be written to the input file, and whether the value should include a decimal point.

Example

```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> TEMPLATE: 1
              input.tpl       input.txt

         >>>> INSTRUCTION: 2
              cost.ins        cost.out
              pump.ins        pump.out

         >>>> EXECUTABLE: pumpcost.bat
         >>>> PRECISION : SINGLE
         >>>> DECPOINT  : POINT
         <<<<
     <<<
  <<
```

See Also

```
>> PEST (o), PEST (p)
```

Command

```
>>>> PHASE phase_name/: iphase
```

Parent Command

```
>>> ELEMENT
>>> SOURCE (o)
```

Subcommand

-

Description

This command identifies a phase either by its name (*phase_name*) or the phase number (*iphase*). A list of allowable phase names for the given EOS module can be obtained from the header of the iTOUGH2 output file.

Example

```
> OBSERVATION
  >> CONCENTRATION
     >>> ELEMENT: ZZZ99
         >>>> ANNOTATION: TCE concentration
         >>>> COMPONENT No.: 3
         >>>> dissolved in LIQUID PHASE
         >>>> DATA on FILE: tce.dat
         >>>> standard DEVIATION: 1.0E-6
         <<<<
     <<<
  <<
```

See Also

```
>>>> COMPONENT
```

Command
>>>> PICK: *npick*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command identifies the number of data points being skipped when reading from a long data file. Only every *npick* data point is read. The default is *npick*=1, i.e., every data point is accepted.

Example
```
> OBSERVATION
  >> TEMPERATURE
     >>> ELEMENTS: ELM_10 + 4
         >>>> SKIP: 3 lines before reading data
         >>>> PICK only every : 10 data point
         >>>> DATA on file: temp.log
         >>>> standard DEVIATION: 0.5 degrees C
         <<<<
     <<<
  <<
```

See Also
>>>> COLUMN, >>>> DATA, >>>> FORMAT, >>>> SET (o), >>>> SKIP

Command
`>>> PLOTFILE: format (LIST)`

Parent Command
`>> OUTPUT`

Subcommand
-

Description
(synonym for command `>>> FORMAT (o)`)

Example
(see command `>>> FORMAT (o)`)

See Also
`>>> FORMAT (o)`

Command
>>> PLOTTING: *niter*

Parent Command
>> OUTPUT

Subcommand
-

Description
By default, the plot file contains the observed data (interpolated at the calibration points), as well as the system response calculated with the initial and final parameter set. Additional intermediate curves can be requested for visualizing the optimization process. Curves are generated for every multiple of *niter* iTOUGH2 iterations.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> PLOTTING: 1 (plots the calculated system response
                       after each iTOUGH2 iteration)
     <<<
  <<
```

See Also
-

Command
>>>> POLYNOM: *idegree* (*time_unit*)

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
Represents observed data by a polynomial of degree ***idegree***:

$$z^*(t) = \sum_{i=0}^{\text{idegree}} \mathrm{A_i} \cdot t^i$$

where t denotes time in ***time_units*** (default is seconds), and $\mathrm{A_i}$ are the ***idegree***+1 coefficients which are read in free format on the lines following the command line.

Example
```
> OBSERVATION
  >> LIQUID FLOW
     >>> CONNECTION: A1__1 A1__2
         >>>> TIME SHIFT: 5 MINUTES
         >>>> conversion FACTOR: 1.6667E-02 [kg/min] - [kg/sec]
         >>>> POLYNOM of degree: 1 (linear function), MINUTES
              2.34  -0.03
               A0     A1  = coefficients of linear regression
                            Through data given in [min] and
                            [kg/min]
         >>>> standard DEVIATION: 0.01 kg/min
         <<<<
     <<<
  <<
```

See Also
>>>> DATA, >>>> USER

Command
```
>> POROSITY
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter porosity (TOUGH2 variable *POR(NMAT)*). This parameter refers to a rock type.
Note that the porosity specified in TOUGH2 block ROCKS (variable *POR(NMAT)*) may be overwritten by non-zero values given in block INCON (variable *PORX*). However, porosity values are not overwritten if the element belongs to a rock type for which porosity is a parameter to be estimated, unless a negative value for *PORX* is provided.

Example
```
> PARAMETER
  >> POROSITY
     >>> ROCK type               :  CLAY1
         >>>> VALUE
         >>>> PRIOR information :  0.28
         >>>> standard DEVIATION:  0.05
         >>>> RANGE              :  0.10 0.50
         >>>> PERTURB            : -0.01
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>> POSTERIORI
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
The estimated error variance s_0^2 represents the variance of the mean weighted residual and is thus a measure of goodness-of-fit:

$$s_0^2 = \frac{\mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r}}{m-n}$$

The *a posteriori* error variance s_0^2 or *a priori* error variance σ_0^2 is used in the subsequent error analysis. For example, the covariance matrix of the estimated parameters, $\mathbf{C}_{pp}$, is directly proportional to the scalar s_0^2 or σ_0^2, respectively. Note that if the residuals are consistent with the distributional assumption about the measurement errors (i.e., matrix $\mathbf{C}_{zz}$), then the estimated error variance assumes a value close to one.
The user must decide whether the error analysis should be based on the *a posteriori* or *a priori* error variance. The decision can also be delegated to the Fisher Model Test (see command `>>> FISHER`). iTOUGH2 uses the *a posteriori* error variance s_0^2 by default.

Example
```
> COMPUTATION
  >> ERROR analysis
     >>> based on A POSTERIORI error variance
     <<<
```

See Also
```
>>> FISHER, >>> PRIORI
```

Command
>>>> POTENTIAL

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
A data-worth analysis examines the value of adding potential observations to an existing calibration data set. Such observations are indicated by command >>>> POTENTIAL. Note that (actually or presumably) existing data sets are also included in the data-worth analysis; their value is examined by removing them from the calibration data set.

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: ELM99
         >>>> POTENTIAL observation...
              (to be added during data-worth analysis)
         >>>> NO DATA available
         >>>> DEVIATION: 1.E5 Pa
         <<<<
      <<<
  <<
```

See Also
>>>> VARIATION, >>> WORTH (op)

Command
```
>>>> PRECISION: SINGLE/DOUBLE
```

Parent Command
```
>>> PEST
```

Subcommand
-

Description
iTOUGH2 capabilities can be applied to non-TOUGH2 models using the PEST interface [*Doherty*, 2002]. The `>>>> PRECISION` command determines whether single or double precision protocol is to be observed in writing parameter values. Unless a parameter space is greater than 13 characters in width it has no bearing on the precision with which a parameter value is written to a model input file, as this is determined by the width of the parameter space. If keyword `SINGLE` is selected, exponents are represented by the letter "e"; also if a parameter space is greater than 13 characters in width, only the last 13 spaces are used in writing the number representing the parameter value, any remaining characters within the parameter space being left blank. If keyword `DOUBLE` is selected, up to 23 characters can be used to represent a number and the letter "d" is used to represent exponents; also, the double-precision range of real numbers is available.

Example
```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE  : "precise.exe < imsingle.in"
         >>>> TEMPLATE    : 1
              precisely.tpl  imsingle.in

         >>>> INSTRUCTION : 1
              precisely.ins  nodifference.out

         >>>> DECPOINT: SINGLE
         <<<<
```

See Also
-

Command
`>>>> PREDICTION`

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
A data-worth analysis examines the value of adding potential or removing (actually or presumably) existing observations to or from an existing calibration data set. Data worth is calculated either by the change in the estimation uncertainty (i.e., matrix $\mathbf{C}_{pp}$), or by the change in model prediction uncertainty (i.e., matrix $\mathbf{C}_{\hat{z}\hat{z}}$). Command `>>>> PREDICTION` indicates that the corresponding observation set represents a prediction for the purpose of performing a data-worth analysis (i.e., these observations are *not* used as a calibration point; their uncertainty is used as a criterion to evaluate data worth; command `>>>> POTENTIAL` is used to indicate potential calibration data sets). Command `>>>> DEVIATION` can be used to indicate the relative importance of making an accurate prediction.

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: ELM99
         >>>> PREDICTION
              (used as criterion for evaluating data worth)
         >>>> NO DATA available
         >>>> DEVIATION: 1.E5 Pa
         <<<<
      <<<
  <<
```

See Also
`>>>> POTENTIAL, >>> WORTH (op)`

Command
```
>> PRESSURE (CAPILLARY) (phase_name/PHASE: iphase)
```

Parent Command
```
> OBSERVATION
```

Subcommand
```
>>> ELEMENT
```

Description
This command selects as an observation type the phase pressure of capillary pressure. This observation type refers to one or more elements. The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the header of the iTOUGH2 output file. They can be specified either on the command line or using subcommand `>>>> PHASE`. If no phase is specified, iTOUGH2 takes the pressure of the first phase, which is usually the reference pressure. The pressure of a given phase is calculated as the sum of the reference pressure and the corresponding capillary pressure.
In a two-phase system, there is only one capillary pressure. The capillary pressure can be selected using keyword `CAPILLARY`. In a three-phase system (e.g., gas, NAPL, aqueous phase), there are two capillary pressures, i.e., the capillary pressure between the NAPL and the gas phase, and between the aqueous and the gas phase, which is the reference phase. The name of the wetting phase must be provided to identify the particular capillary pressure in the three-phase system.

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: AA__1 + 56
         >>>> ANNOTATION    : Mean gas pressure
         >>>> take AVERAGE of pressures in all 57 elements
         >>>> DATA in [MINUTES] on FILE: pres.dat
         >>>> VARIANCE      :   1E6 [Pa^2]
         <<<<
     <<<
  >> NAPL CAPILLARY PRESSURE
     >>> ELEMENT: BB__1
         >>>> take LOGARITHM of
         >>>> ABSOLUTE Pco
         >>>> DATA on FILE : Pco.dat
         >>>> DEVIATION    : 1 log cycle
         <<<<
     <<<
```

See Also
```
>> DRAWDOWN
```

Command
`>>> PRINTOUT: level`

Parent Command
`>> OUTPUT`

Subcommand
-

Description
Specifies the amount of printout in the iTOUGH2 output file. The default level is 5.

level	Amount of printout
0	Same as 1; also suppress reprinting of summary statistics
1	Same as 2; also suppress summary statistics
2	Same as 3; also suppress residual analysis
3	Same as 4: also suppress sensitivity analysis
4	Suppress eigenanalysis, direct correlations, correlation chart
5	Printout of all relevant iTOUGH2 results (default)
6	Add printout of objective function for each forward run (see `>>> OBJECTIVE`)
7	Add printout of residuals for each iteration (see also `>>> RESIDUAL`)
8	Add printout of Jacobian for each iteration (see also `>>> JACOBIAN`)
9	Exhaustive printout

(The amount of printout in the TOUGH2 output file is selected through variable `KDATA`; see *Finsterle* [2015]).

Example
```
> COMPUTATION
  >> OUTPUT
     >>> reduce PRINTOUT to level: 3 ...
     >>> ... but also print OBJECTIVE function for each run
     <<<
  <<
```

See Also
-

Command
>>>> PRIOR: *prior_info*

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command provides the prior information of the parameter to be estimated. If command >>>> PRIOR is omitted, the value from the TOUGH2 input file is taken. Prior information is only effective if a standard deviation is provided (see command >>>> DEVIATION (p)) in which case the difference between the parameter estimate and its prior value is penalized in the objective function. The starting point for the optimization is given through commands >> GUESS and >>>> GUESS.
If command >>>> LOGARITHM is present, the prior information is the logarithm of the parameter. Similarly, if command >>>> FACTOR is present, the prior information should be a multiplication factor (default is 1.0).

Example
```
> PARAMETER
  >> ABSOLUTE permeability
     >>> ROCK type : SAND1
         >>>> LOGARITHM
         >>>> PRIOR information : -12.0
         >>>> standard DEVIATION:   1.0 order of magnitude
         <<<<
     >>> ROCK types: CLAY1 CLAY2 CLAY3 BOUND
         >>>> FACTOR
         >>>> initial GUESS     :   1.0
         >>>> is not WEIGHTed   :   0.0  (default)
         <<<<
     <<<

  >> GUESS, i.e., starting point for optimization
     1  -13.0
  <<
```

See Also
>> GUESS, >>>> GUESS, >>>> DEVIATION, >>>> VARIATION

Command
```
>>> PRIORI
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
The estimated error variance s_0^2 represents the variance of the mean weighted residual and is thus a measure of goodness-of-fit:

$$s_0^2 = \frac{\mathbf{r}^T \mathbf{C}_{zz}^{-1} \mathbf{r}}{m-n}$$

The *a posteriori* error variance s_0^2 or *a priori* error variance σ_0^2 is used in the subsequent error analysis. For example, the covariance matrix of the estimated parameters, $\mathbf{C}_{pp}$, is directly proportional to the scalar s_0^2 or σ_0^2, respectively. Note that if the residuals are consistent with the distributional assumption about the measurement errors (i.e., matrix $\mathbf{C}_{zz}$), then the estimated error variance s_0^2 assumes a value close to one.
The user must decide whether the error analysis should be based on the *a posteriori* or *a priori* error variance. The decision can also be delegated to the Fisher Model Test (see command `>>> FISHER`). iTOUGH2 uses the *a posteriori* error variance s_0^2 by default. However, for design calculations or synthetic inversions, the error analysis should be based on the *a priori* variance σ_0^2 which is 1 by definition.

Example
```
> COMPUTATION
  >> ERROR
     >>> based on A PRIORI error variance
     <<<
```

See Also
```
>>> FISHER, >>> POSTERIORI
```

Command
`>> PRODUCTION (`*`phase_name`*`/PHASE: `*`iphase`*`)`

Parent Command
`> OBSERVATION`

Subcommand
`>>> SINK`

Description
(synonym for command `>> GENERATION`)

Example
(see command `>> GENERATION`)

See Also
`>> GENERATION`

Command
```
>> PRODUCTIVITY INDEX
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> SOURCE
```

Description
This command selects the productivity index for wells on deliverability (TOUGH2 variable *GX*) as a parameter to be estimated. This parameter refers to a sink/source code name. The generation type must be DELV.

Example
```
> PARAMETER
  >> PRODUCTIVITY INDEX
     >>> SINK: WEL_1
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>> PUMPING RATIO
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> SOURCE
```

Description
This command selects as a parameter the pumping ratios of all wells belonging to the same well group. This option can be used to determine the distribution of generation rates among a group of wells, e.g., to determine the optimum pumping strategy for a cleanup operation. For example, the total extraction rate from a system of wells is often limited by the treatment capacity. However, extraction rates of individual wells can be adjusted, optimizing the efficiency of the cleanup operation. A well group consists of all sinks/sources with the same code name (TOUGH2 variable *SL*). The sum of the constant generation rates (TOUGH2 variable *GX*) of all wells within the same well group remains constant, but individual rates are adjusted. It is imperative to estimate as many pumping ratios as the number n of wells within the well group of interest. After optimization, the actual generation rate for well i is calculated from the prescribed total pumping rate q_{tot} and the estimated pumping ratio a_i

$$q_i = q_{tot} \frac{a_i}{\sum_{j=1}^{n} a_j}$$

Example
```
> PARAMETER
  >> PUMPING RATIO
     >>> SOURCE: EXT_1
         >>>> GUESS     : 0.25
         <<<<
     >>> SOURCE: EXT_2
         >>>> GUESS     : 0.5
         <<<<
     >>> SOURCE: EXT_3
         >>>> GUESS     : 0.25
         <<<<
     <<<
  <<
```

See Also
-

Command

```
>>> PVM: nhosts (LEVENBERG: nprocslm / JACOBIAN)
                (SLEEP    : isleep)
                (FILE     : node-file)
HOST1PVM      hostname_1
HOST2PVM      hostname_2
...           ...
HOSTnhostsPVM hostname_nhosts
```

Parent Command

```
>> OPTION
```

Subcommand

-

Description

The number of hosts ***nhosts*** is provided on the command line, followed by ***nhosts*** lines containing the keyword `*HOST`***ihost***`PVM*` (***ihost*** = 1, ... , ***nhosts***) and the name of the host or nodes on a host. The wild card `*` must be a unique identifier if more than one iTOUGH2-PVM applications are run simultaneously. A host (especially a multiprocessor, multi-core machine) may be named several times in the list of hosts. The parent process must not be included in the list.

The nodes can also be provided on an external file (keyword FILE: ***node-file***). This latter option is useful if iTOUGH2 is started using a scheduler on a Linux cluster, i.e., when the nodes are determined dynamically. The names of the available nodes can be written to a file ***node-file*** and read in by iTOUGH2. No list of hosts is required in this case; however, *nhosts* lines with incrementing HOST*ihost*PVM still need to be provided.

If the Levenberg-Marquardt algorithm is used, trial parameter sets obtained with different Levenberg parameters can be evaluated in parallel on ***nrocslm*** processors using keyword `LEVENBERG`. By default, ***nprocslm***=min(6, ***nprocs***). Keyword `JACOBIAN` instructs iTOUGH2-PVM to only evaluate the Jacobian in parallel, i.e., ***nprocslm*** = 1.

The parent process can be suspended for *isleep* seconds each time it checks for incoming residuals (default: ***isleep*** = 1).

More details bout iTOUGH2-PVM can be found in the report *Finsterle* [1998].

Example

Example 1: Heterogeneous cluster of Unix workstations:

```
> COMPUTATION
  >> OPTION
     >>> use LEVENBERG-MARQUARDT minimization algorithm
     >>> PVM: 5 (parallelize JACOBIAN only, SLEEP for : 1 sec)
         HOST1PVM  presto.lbl.gov
         HOST2PVM  hydra.lbl.gov
         HOST3PVM  hydra.lbl.gov
         HOST4PVM  aqua.eth.edu
         HOST5PVM  telos
     <<<
  <<
```

Example 2: Linux cluster; run through scheduler

```
> COMPUTATION
  >> OPTION
     >>> use LEVENBERG-MARQUARDT minimization algorithm
     >>> PVM: 5 nodes, read from FILE: PBS-NODES
         HOST1PVMbatch
         HOST2PVMbatch
         HOST3PVMbatch
         HOST4PVMbatch
         HOST5PVMbatch
     <<<
  <<
```

See Also
Finsterle [1998]

Command
```
>>> QUADRATIC-LINEAR: c
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects a quadratic-linear robust estimator. Given this estimator, the objective function to be minimized is a combination of least-squares for small residuals and the first norm for residuals larger than *c*-times the prior standard deviation:

$$S = \sum_{i=1}^{m} \gamma(y_i)$$

where

$$\gamma(y_i) = \begin{cases} y_i^2 & \text{for } |y_i| \le c \\ c(2|y_i| - c) & \text{for } |y_i| > c \end{cases}$$

with

$$y_i = \frac{r_i}{\sigma_i}$$

This objective function does not correspond to a standard probability density function. It has the general characteristic that the weight given individual residuals first increases quadratically with deviation, then only linearly to reduce the impact of outliers.
For $c \to \infty$, the estimator is identical to least-squares; for $c \to 0$, it approaches the L_1-estimator.
Note that this objective function is minimized using the standard Levenberg-Marquardt algorithm which is designed for a quadratic objective function. Since the function is quadratic for $y_i \le c$, the Levenberg-Marquardt algorithm is usually quite efficient.

Example
```
> COMPUTATION
  >> OPTION
     >>> use a QUADRATIC-LINEAR robust estimator with c : 1.0
     <<<
  <<
```

See Also
```
>>> ANDREWS, >>> CAUCHY, >>> L1-ESTIMATOR, >>> LEAST-SQUARES
```

Command
>>>> RANGE: *lower upper*
or
>>>> BOUND: *lower upper*

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command sets the admissible parameter range. It provides *lower* and *upper* bounds of the parameter. During the optimization process, iTOUGH2 may suggest parameter values that are physically not valid (e.g., negative porosity) or not reasonable (e.g., very high permeability). Limiting the admissible range for the values a parameter can assume prevents the simulation from stopping due to an unphysical or unreasonable parameter value. However, it is strongly suggested not to specify a narrow parameter range about the initial guess. The range should reflect physical bounds, and the expected variation of the parameter. If prior knowledge suggests that a certain parameter varies only slightly about the initial guess, this information should enter the inversion as a standard deviation associated with the initial guess, i.e. prior information (see command >>>> DEVIATION (p)).
A parameter tends to greatly vary, potentially hitting its lower or upper bound, if (i) a systematic error is present, (ii) the initial guess is far away from the best estimate, (iii) the parameter is not sensitive, or (iv) the parameter is highly correlated to more sensitive parameters. The final parameter set should not contain parameters at their lower or upper bounds. If some of the estimated parameters are at the bounds, it is suggested to carefully examine the four potential reasons mentioned above. A new inversion should be performed after corrective actions have been taken.

Example
```
> PARAMETER
  >> POROSITY
     >>> MATERIAL: SANDY
         >>>> PRIOR information : 0.34
         >>>> standard DEVIATION: 0.05
         >>>> admissible RANGE  : 0.01 0.99
         <<<<
     <<<
  <<
```

See Also
>>>> DEVIATION (p), >>>> STEP

Command
```
>> RATE
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> SOURCE
```

Description
This command selects as a parameter the constant generation rate (TOUGH2 variable *GX*) or time-dependent generation rate (TOUGH2 variable *F2(L)* for *LTAB* > 1). This parameter refers to a sink/source code name. Estimating a time-dependent generation rate requires providing index *L* through command `>>>> INDEX`.

Example
```
> PARAMETER
  >> RATE
     >>> SOURCE: WEL_1 + 5
         >>>> ANNOTATION: const. rate
         >>>> LOGARITHM
         <<<<
     >>> SOURCE: INJ_1
         >>>> ANNOTATION: variable rate
         >>>> VALUE
         >>>> INDEX     : 7
         <<<<
     <<<
  <<
```

See Also
```
>> TIME (p)
```

Command
`>>> REDUCTION: max_red`

Parent Command
`>> CONVERGE`

Subcommand
-

Description
By default, a TOUGH2 simulation stops if 20 consecutive time step reductions have occurred without convergence. This command changes the maximum number of allowable time step reductions to *max_red*.
Time step reductions occur if:

(1) the initial time step is too large (see TOUGH2 variable *DELTEN* or *DLT(1)*, and command `>>> ADJUST`),

(2) TOUGH2 parameter *NOITE* is too small,

(3) a failure in the EOS module occurs ,

(4) boundary conditions are changed drastically during the course of the simulation.

Variable *max_red* should only be increased to address problem (4), i.e., when drastic changes in generation rates (TOUGH2 block GENER) are imposed, or if Dirichlet boundary conditions are changed using command `>> RESTART TIME` or through subroutine `USERBC`.

Example
```
> COMPUTATION
  >> CONVERGE
     >>> accept : 30 CONSECUTIVE time step reductions
     <<<
  <<
```

See Also
`>> RESTART TIME, >>> ADJUST, >>> CONSECUTIVE`

Command
`>> REGION (PERMEABILITY, SINK/SOURCE, OBSERVATION)`

Parent Command
`> PARAMETER`

Subcommand
`>>> MATERIAL` (if keyword `PERMEABILITY`)
`>>> MODEL` (if keyword `OBSERVATION`)
`>>> SOURCE` (if keyword `SINK/SOURCE`)

Description
This command selects as a parameter the location and size of a region, in which either properties (i.e., permeability and porosity), a sink or source is provided, or observations are taken (for monitoring system design); see *Finsterle* [2015] for specifying property, injection or production regions. The specific geometric parameter *XREGION(`i`)* defined below is identified by command `>>>> INDEX: i`. Its meaning depends on the region geometry as defined by parameter *IREGGEOM*.

IREGGEOM	*XREGION(`i`)*								
	1	2	3	4	5	6	7	8	9
1 (box)	X_{min}	Y_{min}	Z_{min}	X_{max}	Y_{max}	Z_{max}	aximuth	dip	plunge
2 (ellipsoid)	X_{center}	Y_{center}	Z_{center}	$X_{semi-axis}$	$Y_{semi-axis}$	$Z_{semi-axis}$	aximuth	dip	plunge
3 (cylinder)	X_{start}	Y_{start}	Z_{start}	X_{end}	Y_{end}	Z_{end}	radius	-	-
4 (cube)	X_{center}	Y_{center}	Z_{center}	$X_{half-length}$	$Z_{half-length}$	$Z_{half-length}$	aximuth	dip	plunge

If `i` is negative, the estimated parameter is added to the initial values of the parameters in the above table; this feature is useful, for example, to shift multiple wells or discrete fault zones described by permeability regions.

If keyword `OBSERVATION` is present, the parameter refers to the region where observations are taken. Making the location of a potential measurement and adjustable parameter may be useful for assessing the impact of measurement location uncertainty and for optimal monitoring system design. If this option is used, the annotation of the parameter has to be exactly the same as that of the corresponding data set in the `>> OBSERVATION` block. Moreover, the observation location may not be defined by a list of elements or connections, but by providing the corrdinates of the region.

Example

```
> PARAMETER
  >> SOURCE REGIONs
     >>> SOURCE: XYZ_1 +9
         >>>> ANNOTATION: shift all wells N-S
         >>>> INDEX     : -2  -5
         >>>> VALUE
         >>>> GUESS     : 10.0 [m]
         >>>> RANGE     : -50.0 50.0
         <<<<
     <<<

  >> OBSERVATION REGION
     >>> MODEL
         >>>> ANNOTATION: concentration
         >>>> INDEX     : -3 (sampling depth)
         >>>> VALUE
         >>>> RANGE     : -30 -20
         <<<<
     <<<
  <<

> OBSREVATION
  >> CONCENTRATION
     >>> ELEMENT CUBE COORDINATES: 20.  45. -25.  .5  .5  2.
         >>>> ANNOTATION: concentration
         >>>> NO DATA
         >>>> DEVIATION: 1.0E-6
         <<<<
     <<<
  <<
```

See Also

-

Command
```
>> REGRESSION
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects the autoregressive coefficient ρ as the parameter to be adjusted or estimated. To partially account for time correlations, a first-order autogregressive time series model (AR1) can be applied to the residuals:

$$\tilde{r}_i = \begin{cases} r_1\sqrt{(1-\rho^2)} & \text{if } i = 1 \\ r \ -\rho \cdot r_{i-1} & \text{if } i = 2,\ldots,m \end{cases}$$

Autoregression is performed on the time-series residuals of a data set in block `>> OBSERVATION`, which must contain the command `>>>> REGRESSION: rho`. The data set is identified by its sequence number through subcommand `>>>> INDEX: iset`.

Example
```
> PARAMETER
  >> REGRESSION
     >>> MODEL
         >>>> INDEX: 1
         >>>> VALUE
         >>>> RANGE: -1 1
         >>>> VARIATION: 0.1
         <<<<
     <<<
  <<

> OBSERVATION
  >> PRESSURE
     >>> ELEMENT                  : A1125
         >>>> Read DATA from FILE : pres.col
         >>>> standard DEVIATION  : 200.0 Pa
         >>>> REGRESSION          : 0.5
         <<<<
```

See Also
```
>>>> REGRESSION, >> SHIFT, >> DRIFT
```

Command
>>>> REGRESSION

Parent Command
any third-level command in block > OBSERVATION

Subcommand
none

Description
This command sets the autoregressive coefficient ρ for a data set. To partially account for time correlations, a first-order autogregressive time series model (AR1) can be applied to the residuals:

$$\tilde{r}_i = \begin{cases} r_1\sqrt{(1-\rho^2)} & \text{if } i = 1 \\ r \ -\rho \cdot r_{i-1} & \text{if } i = 2,\ldots,m \end{cases}$$

The first-order autoregressive should be applied to time series only if the sampling interval is approximately constant. The autoregression coefficient *rho* can be estimated using command >> REGRESSION.

Example

```
> OBSERVATION

  >> PRESSURE
     >>> ELEMENT                  : A1125
         >>>> Read DATA from FILE : pres.col
         >>>> standard DEVIATION  : 200.0 Pa
         >>>> REGRESSION          : 0.5
         <<<<
     <<<
```

See Also
>>>> DETREND, >> REGRESSION, >>>> SHIFT

Command
>> REGULARIZATION (FILE: *file_name*) (BETA: *beta*)

Parent Command
> PARAMETER
> OBSERVATION

Subcommand
-

Description
This command identifies pairs of parameters that are used to calculate a smoothness regularization term. Weighted differences between the two estimated values are added to the objective function. The pairs are identified by the ordering number in the PARAMETER block. A reagularization weight can be given for the regularization term as a whole (keyword BETA; default: 1.0). Moreover, for each pair, a standard deviation can be provided (default: 1.0); for smoothing, this standard deviation is typically related to the separation distance between the two parameter points. Note, however, that this regularization approach is not restricted to spatial smoothing of a field of estimated parameters of the same type, but can be used to constrain differences between parameters of different types (e.g., the residual saturation for the relative permeability and capillary pressure functions).

The information can be given directly in the iTOUGH2 input file, or on a separate file given after keyword FILE). Two integers are read in free format to indicate the parameter pair. Providing a standard deviation is optional.

This command can be given in the PARAMETER block (as it relates to differences in estimated parameter values), or in the OBSERVATION block (as it adds to the objective function as do other residuals between observed and measured values).

Regularization by means of differences between an estimated value and its preferred value is accomplished through command >>>> PRIOR.

Example
```
> PARAMETER
  >> REGULARIZATION, BETA: 100.0
     1   2   0.1541
     1   4   0.2573
     2   5   0.2573
     4   5   0.1541
     5   6   0.1541
     5   8   0.2573
```

See Also
>>>> PRIOR

Command
```
>> RELATIVE
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> DEFAULT
>>> MATERIAL
```

Description
This command selects a parameter of the relative permeability function (TOUGH2 variable *RP(IPAR,NMAT)*) of a certain rock type, or a parameter of the default relative permeability function (TOUGH2 variable *RPD(IPAR)*). Command `>>>> INDEX` must be used to select the parameter index *IPAR*. The physical meaning of the parameter depends on the type of relative permeability function selected in the TOUGH2 input file, variable *IRP* or *IRPD*. The admissible range should be specified explicitly to comply with parameter restrictions (see *Pruess* [1987], Appendix A).

Example
```
> PARAMETER
  >> parameter of RELATIVE permeability function
     >>> DEFAULT
         >>>> ANNOTATION    : Resid. Gas Sat.
         >>>> PARAMETER no. : 2
         >>>> VALUE
         >>>> RANGE         : 0.01 0.99
         <<<<
     >>> MATERIAL: ZONE1 +3
         >>>> ANNOTATION    : IRP=7, RP(1)=m
         >>>> PARAMETER no. : 1
         >>>> FACTOR
         <<<<
     <<<
  <<
```

See Also
```
>> CAPILLARY
```

Command
>>>> RELATIVE: *rel_err* (%) (+ *const_err*) (ADD NOISE)

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command takes a fraction of the observed value as the standard deviation of the measurement error (for more details see command >>>> DEVIATION (o)).
A constant error can be added to the error model.

If keyword ADD NOISE is present, Gaussian noise with zero mean and the given standard deviation is added to the data points. This option may be useful if error-free data were synthetically generated.

Example
```
> OBSERVATION
  >> LIQUID FLOW RATE
     >>> CONNECTION INJ_1 DOM_2
         >>>> DATA on FILE: flow.dat
         >>>> RELATIVE measurement error: 5 % + 0.1
         <<<<
     <<<
  <<
```

See Also
>>>> DEVIATION (o)

Command
```
>>> RESIDUAL
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints the observed and calculated system response as well as the residuals after each iTOUGH2 iteration. By default, the residuals are printed only once at the end of the optimization.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print RESIDUALs after each TOUGH2 simulation.
     <<<
  <<
```

See Also
```
>>> JACOBIAN
```

Command
```
>>> RESOLUTION
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
Prints model (parameter) and data resolution matrices to a file with extension *.res*.

The data resolution matrix indicates how well the model predictions are resolved, i.e., to what extent they are an average of (usually neighboring) observations.
Note that for overdetermined inverse problems, the model resolution matrix is the identity matrix (see `>>> IDENTIFIABILITY`). The traces of the resolutions matrices should be equal to the number of adjustable parameters.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print RESOLUTION matrices
     <<<
  <<
```

See Also
```
>>> IDENTIFIABILITY
```

Command
`>> RESTART TIME: ntime (time_unit) (NEW)`

Parent Command
`> OBSERVATION`

Subcommand
-

Description
This command selects points in time at which step changes are to be made to individual elements or element groups. At this specific point in time, the user can either change the element's volume, porosity, permeability, material type, or any of its primary variables. Such times are termed "restart times" because in standard TOUGH2 one would have to stop the simulation, and restart it after having changed the properties and initial conditions of certain elements.
This option is useful, for example, for modeling well tests, where the condition in the borehole is changed at certain points in time. For the simultaneous calibration of well tests consisting of many test events, it is imperative to model the entire sequence in a single simulation run, i.e., individual events must be connected. Recall that complicated, time-varying boundary conditions can also be programmed into subroutine USERBC.
The specification of a restart time is identical to specifying calibration points (see command `>> TIME (o)`). Instead of calibration times, a restart time *`restart_time`* is indicated. Following the line containing the restart time, elements are defined by a *`name`*, which is an element name, an element number, or a material name ('`?`' and '`*`' in the material name act as wildcards). The name is followed by an integer *`ipv`* (see table below). If *`ipv`* is negative the value is taken as a change from the conditions encountered at the restart time.

abs(ipv)	*value*
0	element volume
$1 \le$ *ipv* $\le$ NEQ*	primary variable No. *ipv*
NEQ + 1	porosity
NEQ + 2	permeability or permeability modifier (see *Finsterle* [2015])
NEQ + 1 + *i*	variable USERX(*i*) (see *Finsterle* [2015])
99	material number
999	phase index (e.g., for TMVOC)

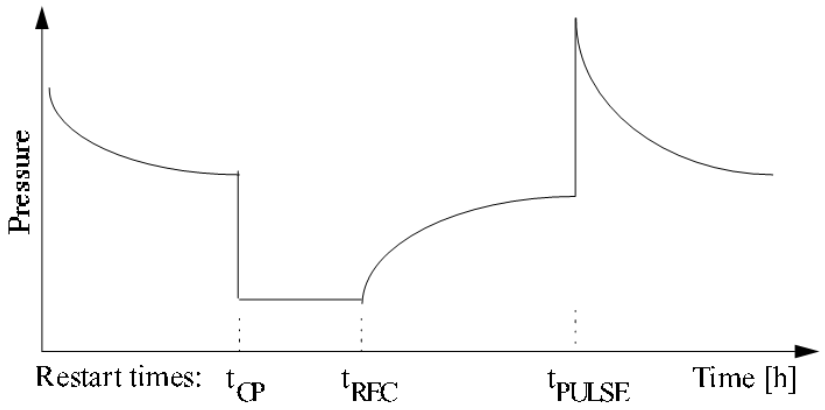
* NEQ is the number of equations solved per grid block.

The general format is as follows:

```
>> RESTART TIME: 1 (time_unit)
   restart_time
   name  ipv  value
   name  ipv  value
   ....  ...  ..
```

Example

This example demonstrates the use of restart times to connect four individual test events into a single simulation. The test sequence starts with a shut-in recovery period, i.e. in the TOUGH2 input file the actual interval volume is assigned to the element representing the borehole. At time t_{CP}, a constant-pressure pumping test is initiated, i.e., the element volume is increased to a very large number, and the prescribed interval pressure is specified as initial conditions. A shut-in pressure recovery period starts at time t_{REC}, modeled by reassigning the actual interval volume. Finally, a pulse test is simulated, assuming that all the gas that potentially accumulates in the borehole interval has been released at time t_{PULSE}, prior to applying a short pressure pulse. The pressure response in the injection well is schematically shown in the figure below, followed by the appropriate iTOUGH2 block with restart times and boundary condition specifications.




```
> OBSERVATION
  >> RESTART TIME: 1    [HOURS]
     2.56               (tCP)
     BOR_1   0  1.0E+50 (large volume for const. pressure b.c.)
     BOR_1   1  1.3E+05 (set constant interval pressure)

  >> RESTART TIME: 1    [HOURS]
     3.51               (tREC)
     BOR_1   0  0.73    (actual volume for shut-in recovery)

  >> RESTART TIME: 1    [HOURS]
     5.44               (tPULSE)
     BOR_1   2  0.00    (prescribe single-phase liquid cond.)
     BOR_1   1  4.85E+5 (pressure pulse)
      |       |   |
    element  ipv vol
```

TOUGH2 stops at each restart time, writes a SAVE file, reads it as an INCON file, changes initial conditions, i.e. updates the primary variables or grid block volumes of the indicated elements, and continues the simulation.

See Also

```
>> TIME (o)
```

Command
`>>> ROCKS: mat_name (mat_name_i ...) (+ iplus)`

Parent Command
any second-level command in block `> PARAMETER` referring to a material name

Subcommand
(synonym for command `>>> MATERIAL`)

Description
(see command `>>> MATERIAL`)

Example
-

See Also
`>>> MATERIAL`

Command

```
>> SATURATION (phase_name/PHASE: iphase)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> ELEMENT
```

Description

This command selects phase saturation as an observation type. This observation type refers to an element. The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the iTOUGH2 header. They can be specified either on the command line or using the subcommand >>>> PHASE.

Example

```
> OBSERVATION
  >> GAS SATURATION
     >>> calculated for ELEMENT: AA__5
         >>>> ANNOTATION: Sg during gas injection
         >>>> DATA  [HOURS]
              0.1  0.01
              0.3  0.13
              0.5  0.34
              0.8  0.59
              1.2  0.72
              1.5  0.81
              2.0  0.86
              2.5  0.89
              5.0  0.91
         >>>> DEVIATION: 0.05
         <<<<
     <<<

  >> SATURATION
     >>> ELEMENT: BB__5
         >>>> NAPL PHASE saturation
         >>>> NO DATA available (i.e., just for plotting)
         >>>> WEIGHT: 1.0E-20   (don't weigh, just plot!)
         <<<<
      <<<
  <<
```

See Also

-

Command
```
>> SCALE
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
```

Description
This command selects as a parameter a grid scaling factor. Nodal distances, interface areas, and gridblock volumes will be scaled accordingly. There are three grid scaling factors, referring to the three directions specifed by TOUGH2 variable *ISOT*. The direction is selected through command `>>>> INDEX`. If all three directions are selected, the mesh is scaled isotropically (see TOUGH2 variable *SCALE*).

Selecting a grid scaling factor as a parameter can be used to design and optimize the horizontal and/or vertical spacing between injection points.

Example
```
> PARAMETER
  >> grid SCALEing factor
     >>> NONE
         >>>> ANNOTATION: horizontal well spacing
         >>>> INDEX     : 1 2
         >>>> GUESS     : 1.0
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>> SCALING
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
>>> SET
```

Description
This command selects as a parameter a constant factor with which the calculated TOUGH2 output will be multiplied. The factor is applied to the output that refers to a specific data set:

$$z = z_{TOUGH\,2} \cdot factor$$

where *factor* is the scaling factor and z_{TOUGH2} is the TOUGH2 output. The result z is compared to the measurement z^* of the corresponding data set.
This option allows correcting for a systematic, but unknown relative error in the data.
The data set is identified by number using command `>>> SET (p)`.
If the factor is known and does not need to be estimated, use command
`>>>> FACTOR (o)`.

Example
```
> PARAMETER
  >> SCALING factor
     >>> SET No. : 1
         >>>> ANNOTATION: correct amplitude
         >>>> GUESS: 1.0
         <<<<
     <<<
  <<
```

See Also
`>> DRIFT, >> LAG, >> SHIFT, >>> SET (p), >>>> FACTOR (o)`

Command
```
>>>> SCHEDULE: beta
```

Parent Command
```
>>> ANNEAL
```

Subcommand
-

Description
This command defines the annealing schedule for Simulated Annealing minimization. The annealing schedule is a function describing the temperature reduction (and thus the probability of accepting an uphill step, see command `>>> ANNEAL`). The temperature is updated after a certain number of successful steps have been performed (see command `>>>> STEP (a)`).
There are two functions available. If ***beta*** is in the range $0 < \beta < 1$ (typically 0.9), the following annealing schedule is used:

$$\tau_k = \beta^k \cdot \tau_0$$

where k is the iteration index, and $\tau_0$is the initial temperature specified by command `>>>> TEMPERATURE`. If ***beta*** is greater than one, the following annealing schedule is invoked:

$$\tau_k = \tau_0 (1 - k / K)^\beta$$

where K is the maximum number of iterations (see command `>>>> ITERATION (a)`).

Example
```
> COMPUTATION
  >> OPTION
     >>> Simulated ANNEALing
         >>>> total number of ITERATIONS: 200
         >>>> initial TEMPERATURE: -0.05
         >>>> update after: 100 STEPS
         >>>> annealing SCHEDULE: 0.95
         <<<<
     <<<
  <<
```

See Also
```
>>>> ITERATION (a), >>>> STEP, >>>> TEMPERATURE
```

Command
```
>> SECONDARY (phase_name/PHASE: iphase) (: ipar)
```

Parent Command
```
> OBSERVATION
```

Subcommand
```
>>> ELEMENT
```

Description
This command selects thermophysical properties, which are so-called "secondary parameters" in TOUGH2, as an observation type. This observation type refers to an element.
The secondary parameter refers to elements of TOUGH2 vector *PAR* (see *Pruess* [1991a], Figure 2). The index *ipar* must be provided in the command line after a colon or through the fourth-level command `>>>> INDEX`. The fluid phase must also be identified. The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the iTOUGH2 header. They can be specified either on the command line or using the fourth-level command `>>>> PHASE`.
The following table lists the thermophysical property addressed by *ipar*. Some of the properties can be selected using an alternative second-level command.

ipar	Secondary Parameter	Alternative Command
1	phase saturation	`SATURATION`
2	relative permeability	–
3	viscosity	–
4	density	–
5	specific enthalpy	–
6	capillary pressure	`PRESSURE`
NB+i	mass fraction of component *i*	`MASS FRACTION`

Example
```
> OBSERVATION
  >> SECONDARY parameter No: 3
     >>> ELEMENT: AA__5
         >>>> ANNOTATION: gas viscosity
         >>>> GAS PHASE
         >>>> NO DATA
         <<<<
     <<<
```

See Also
`>> MASS FRACTION, >> PRESSURE, >> SATURATION, >>>> PHASE, >>>> INDEX`

Command
```
>> SELEC
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MODEL
```

Description
This command selects as a parameter one of the SELEC parameters (TOUGH2 variable *FE(I)*). The physical meaning of the parameter depends on the module being used. The index *I* must be provided through command `>>>> INDEX`.

Example
```
> PARAMETER
  >> SELEC parameter
     >>> MODEL
         >>>> ANNOTATION: Gel Density (EOS11)
         >>>> INDEX     : 2
         >>>> VALUE
         <<<<
     <<<
  <<
```

See Also
-

Command

```
>>> SELECT
>>> SUPER
```

Parent Command

```
>> OPTION
```

Subcommand

```
>>>> CORRELATION
>>>> EIGENVALUE
>>>> IMMOBILIZATION
>>>> ITERATION
>>>> LIST
>>>> SENSITIVITY
>>>> TRUNCATE
```

Description

This command invokes the automatic parameter selection option of iTOUGH2. The parameters defined in block `> PARAMETER` are screened according to selection criteria (see commands `>>>> SENSITIVITY`, `>>>> CORRELATION`) or will be truncated based on the eigenvalues of the Fisher information matrix (see commands `>>>> EIGENVALUE` or `>>>> TRUNCATE`). Only the most sensitive and/or most independent parameters are subjected to the optimization process. If command `>>> SUPER` is used, superparameters will be generated and optimized (see *Tonkin and Doherty* [2005]).

Automatic parameter selection allows one to submit a larger set of parameters to the estimation process. If a parameter is not sensitive enough or too correlated to be estimated from the available data, it is automatically removed from the set of parameters being updated. This makes the inversion faster because fewer parameters must be perturbed for calculating the Jacobian matrix (the full Jacobian is only calculated every few iteration when the selection criteria are reevaluated). The inversion is also more stable.

Due to the non-linearity of the inverse problem at hand, sensitivity coefficients and parameter correlations constantly change during the optimization. Therefore, the selection criteria must be reevaluated from time to time (see command `>>>> ITERATION (s)`), i.e., parameters may be deactivated and reactivated during the course of an inversion.

Example

```
> COMPUTATION
  >> OPTION
     >>> SELECT parameters based on
         >>>> CORRELATION criterion : -3
         >>>> SENSITIVITY criterion : 0.2
         >>>> repeat selection every : 3 rd ITERATION
         <<<<
```

See Also

-

Command
```
>>> SENSITIVITY (LOCAL)
```
or
```
>>> DESIGN
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command makes iTOUGH2 evaluate the sensitivity matrix without performing any optimization. By default, the scaled Jacobian matrix, i.e., the local sensitivity coefficients scaled by the standard deviation of the observation and expected parameter variation, respectively, is printed to the iTOUGH2 output file:

$$\tilde{J}_{ij} = J_{ij} \frac{\sigma_{p_j}}{\sigma_{z_i}} = \frac{\partial z_i}{\partial p_j} \cdot \frac{\sigma_{p_j}}{\sigma_{z_i}}$$

In addition, the unscaled sensitivity coefficients can be printed by invoking subcommand `>>> SENSITIVITY` in block `>>> OUTPUT`.
This information can be used to identify the parameters that most strongly affect the system behavior at actual or potential observation points. Similarly, the relative information content of actual or potential observations, i.e., the contribution of each data point to the solution of the inverse problem can be evaluated.
Based on this command, iTOUGH2 also calculates the covariance matrix of the estimated parameters, i.e., the estimation uncertainty under the assumption that the variances of the residuals are accurately depicted by the prior covariance matrix $\mathbf{C}_{zz}$. This information along with the global sensitivity measures (sums of absolute sensitivity coefficients) can be used to optimize the design of an experiment.
It is recommended to use a relatively large perturbation factor (see command `>>> PERTURB)`, possibly in combination with centered finite difference quotients (see command `>>> CENTERED)` for the purpose of sensitivity analysis.

Example
```
> COMPUTATION
  >> OPTION
     >>> perform a SENSITIVITY analysis for test DESIGN
     <<<
```

See Also
`>>> CENTERED, >>> PERTURB, >>> SENSITIVITY MORRIS, >>> SENSITIVITY SALTELLI/SOBOL, >>> TORNADO`

Command
```
>>> SENSITIVITY MORRIS
```

Parent Command
```
>> OPTION
```

Subcommand
```
>>>> PARTITION
>>>> PATH
```

Description
This command performs a global sensitivity analysis using the Morris method. The parameter range is partitioned uniformly into a discrete set of values. Startig from a randomly chosen initial set, parameters perturbed one at a time in random order. Multiple random paths *j* in the parameter space can be evaluated for a total of $j(n+1)$ simulations, based on which elementary effects and variances are calculated, reflecting relative parameter importance as well as linearity and correlation. For details, see *Morris* [1991] and *Wainwright and Finsterle* [2015].

Example
```
> COMPUTATION
  >> OPTION
     >>> perform a MORRIS AOT SENSITIVITY analysis
         >>>> number of PATHs: 5
         >>>> PARTITION parameter range into: 10 segments
         <<<<
     <<<
  <<
```

See Also
>>> SENSITIVITY (LOCAL), >>> SENSITIVITY SALTELLI/SOBOL, >>> TORNADO

Command
```
>>> SENSITIVITY SALTELLI/SOBOL
```

Parent Command
```
>> OPTION
```

Subcommand
```
>>>> SAMPLES
```

Description
This command performs a global sensitivity analysis using the method of Sobol' as modified by Saltelli. For details, see *Saltelli et al*. [2008] and *Wainwright and Finsterle* [2015].

Example
```
> COMPUTATION
  >> OPTION
     >>> perform SOBOL global SENSITIVITY analysis
         >>>> number of SAMPLEs: 10000
         <<<<
     <<<
  <<
```

See Also
>>> SENSITIVITY (LOCAL), >>> SENSITIVITY MORRIS, >>> TORNADO

Command
```
>>> SENSITIVITY
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints the sensitivity matrix to the iTOUGH2 output file. The elements of the sensitivity matrix are the partial derivatives of the system response at the observation points with respect to the parameters:

$$J_{ij} = \frac{\partial z_i}{\partial p_j}$$

By default, iTOUGH2 prints the scaled Jacobian matrix, the elements of which are the sensitivity coefficients scaled by the ratio of the standard deviation of the observation and expected parameter variation:

$$\tilde{J}_{ij} = J_{ij} \frac{\sigma_{p_j}}{\sigma_{z_i}} = \frac{\partial z_i}{\partial p_j} \cdot \frac{\sigma_{p_j}}{\sigma_{z_i}}$$

Example
```
> COMPUTATION
  >> OUTPUT
     >>> print SENSITIVITY matrix in addition to the
         scaled Jacobian
     <<<
  <<
```

See Also
```
>>> IDENTIFABILITY
```

Command
`>>>> SENSITIVITY: (-)rsens`

Parent Command
`>>> SELECT`

Subcommand
-

Description
This command defines one of the criteria used for automatic parameter selection. It examines the potential of a parameter to reduce the objective function:

$$\delta = |\partial S|$$

Here, ∂S is the change of the objective function if the parameter is perturbed by a certain, small value (see command `>>> PERTURB`).

Normalizing to the maximum value $\delta_{\max}$ yields the selection criterion ω:

$$\omega = \frac{\delta}{\delta_{\max}} 0 < \omega \le 1$$

Those parameters with an ω-value larger than $|rsens|$, i.e., the most sensitive parameters, are selected. Parameters which are unlikely to significantly reduce the objective function are (temporarily) excluded from the optimization process.
If a negative value is given for $rsens$, the selection criterion is relaxed with each iteration k, and reaches zero for the last iteration max_iter, i.e., all parameters specified in block `> PARAMETER` are selected for the final step.

$$rsens_k = |rsens| \cdot \left(1 - \frac{k}{max\_iter}\right)$$

Example
```
> COMPUTATION
  >> OPTION
     >>> SELECT parameter automatically every
         >>>> : 3 ITERATIONS
         >>>> based on the SENSITIVITY criterion, rsens : -0.10
         <<<<
      <<<
  <<
```

See Also
`>>>> ITERATION (s), >>>> CORRELATION`

Command
>>> SET: *iset*

Parent Command
>> DRIFT
>> LAG
>> SCALING
>> SHIFT

Subcommand
any fourth-level command in block > PARAMETER

Description
The parent command of this command identifies a parameter that refers to a data set, i.e., it manipulates the observed data. The data set is identified by its ordering number *iset* as specified in block > OBSERVATION. Alternatively, *iset* can be read using subcommand >>>> INDEX; the latter command must be used if multiple sets are selected.

Example
```
> PARAMETER

Estimate coefficients of regression dz=A+B*time to correct
flowmeter data.  The flowmeter data are provided as data set
No. 2, 3, and 4 in the OBSERVATION block.  All three data sets
exhibit a constant offset to be estimated.  Data set No. 3 also
shows a drift.

  >> SHIFT
     >>> NONE (multiple sets will be selected)
         >>>> INDEX     : 2 -4
         >>>> ANNOTATION: constant A (set 2, 3 and 4)
         >>>> GUESS     : 4.0E-6 [kg/sec]
         <<<<
     <<<

  >> DRIFT
     >>> data SET No.   : 3
         >>>> ANNOTATION: coefficient B (set 3 only)
         >>>> GUESS     : 1.0E-9 [kg/sec/sec]
         <<<<
     <<<
  <<
```

See Also
-

Command
>>>> SET: *iset*

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
Multiple data sets can be stored on a single file, separated by a single line containing characters. This command is used to select the *iset*'th data set on a data file. By default, data are read from the first set following the header lines (see command >>>> HEADER).

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: AAA99
         >>>> ANNOTATION: Pres. well BBB
         >>>> conversion FACTOR: 1.0E5
         >>>> skip: 3 HEADER lines, then
         >>>> select: 2 nd SET on
         >>>> COLUMNS: 2 3
         >>>> DATA FILE: pres.dat [HOURS]
         >>>> RELATIVE error: 0.05
         <<<<
     <<<
```

The content of file *pres.dat* is:

```
   1 This is file pres.dat
Line Time     Pressure
   3 Data set 1: Pressure in borehole AAA
   4  0.0     1.000
   5  1.0     1.134
   6  2.0     1.495
   7 Data set 2: Pressure in borehole BBB
   8  0.0     1.051
   9  1.0     1.433
  10  2.0     1.874
  11  3.0     2.431
  ..  ...     .....
```

See Also
>>>> HEADER

Command
```
>> SHIFT
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> NONE
>>> SET
```

Description
This command selects as a parameter a constant used to shift the calculated TOUGH2 output. The constant is added to the output that refers to a specific data set:

$$z = z_{TOUGH2} + shift$$

where *shift* is the constant added to the calculated TOUGH2 output z_{TOUGH2}. The result z is compared to the measurement z^* of the corresponding data set.
This option allows removal of a constant from the data. For example, if matching relative pressure data that fluctuate around an unknown mean, or if a flowmeter exhibits an unknown offset that needs to be estimated, the shift parameter being estimated is the mean or offset, respectively.
A non-zero value must be provided as an initial guess through the iTOUGH2 input file using command `>>>> GUESS`.
The data set is identified by number using command `>>> SET (p)` or command `>>>> INDEX (p)`.
If the constant is known, i.e. does not need to be estimated, use command `>>>> SHIFT`, a subcommand of `> OBSERVATION`, to shift the data.

Example
```
> PARAMETER
  >> SHIFT
     >>> SET No. : 1
         >>>> ANNOTATION: atmos. pres.
         >>>> GUESS: -1.0E5 [Pa] subtract atmospheric pressure
         <<<<
     >>> NONE
         >>>> ANNOTATION: offset
         >>>> INDEX     : 2 3 4 one constant for all data sets
         >>>> GUESS     : 4.0E-6 [kg/sec]
         <<<<
     <<<
  <<
```

See Also
```
>> DRIFT, >> SCALING, >> LAG, >>> SET (p), >>>> SHIFT
```

Command
>>>> SHIFT: *shift* (TIME (*time_unit*))

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
This command shifts data values or observation times by a known, constant value ***shift***. The effect of this command is a direct data manipulation, which must be distinguished from estimating an unknown constant that shifts the model output (see command >> SHIFT). If keyword TIME is present on the command line, the constant ***shift*** is added to all observation times of the corresponding data set. This is useful to adjust the reference time of the data record if it is different from the starting time of the simulation (TOUGH2 variable ***TSTART***). The time unit can be selected using one of the ***time_unit*** keywords. If keyword TIME is not present, *shift* is added to all observed values. The units of ***shift*** must match those of the observed data (see command >>>> FACTOR).

Example
```
> OBSERVATION
  >> PRESSURE
     >>> ELEMENT: GUG_0
         >>>> ANNOTATION: rel. pres.
         >>>> conversion FACTOR: 1000.0 [kPa] - [Pa]
         >>>> SHIFT: 100.0 [kPa] add atmospheric pressure
         >>>> SHIFT TIMES as well by: -1.25 DAYS
         >>>> relative pressure DATA in [HOURS] and [kPa]:
              30.0  1.314
              30.5  1.789
              31.0  2.113
              ....  .....
         >>>> standard DEVIATION: 0.5 [kPa]
         <<<<
     <<<
  <<
```

See Also
>> SHIFT, >>>> FACTOR

Command
```
>>> SIGNAL
```

Parent Command
```
>> STOP
```

Subcommand
-

Description
This command bypasses the signal handler that intercepts signals sent by the user (ctrl-C on a PC, or those sent by script *kit* on unix systems). Bypassing the signal handler may be needed to debug the code, so that events throwing a signal can be located.

Example
```
> COMPUTATION
  >> STOP
     >>> bypass SIGNAL handler
     <<<
  <<
```

See Also
-

Command
```
>>> SIMPLEX
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command selects the downhill simplex method to minimize the objective function. The downhill simplex method does not calculate derivatives and can therefore be used for discontinuous objective functions. The initial simplex is defined by the initial parameter set $\mathbf{p}_0$ and n additional points given by:

$$\mathbf{p}_i = \mathbf{p}_0 + \sigma_i \cdot \mathbf{e}_i \quad i = 2,...,(n+1)$$

where σ_i is the parameter variation given by commands `>>>> VARIATION` or `>>>> DEVIATION (p)`, and the $\mathbf{e}_i$'s are n unit vectors along the parameter axis.
At the end of minimization by means of the simplex algorithm, the Jacobian matrix is evaluated to allow for a standard error analysis.
The simplex algorithm is most useful in connection with the L_1-estimator and discontinuous cost functions.

Example
```
> COMPUTATION
  >> OPTION
     >>> use SIMPLEX algorithm to minimize objective function
     <<<
  <<
```

See Also
```
>>> ANNEAL, >>> GAUSS-NEWTON, >>> GRID SEARCH,
>>> LEVENBERG-MARQUARDT
```

Command
>>> SIMULATION: *mtough2*

Parent Command
>> CONVERGE

Subcommand
-

Description
This command limits the number of TOUGH2 simulations performed during an inversion. A rough estimate of the total number of TOUGH2 simulations during an inversion can be made given the number of iterations requested (see command >>> ITERATION) and the type of finite difference quotients calculated (see commands >>> FORWARD and >>> CENTERED). However, there is an unknown number of unsuccessful steps that cannot be accounted for.
The maximum number of TOUGH2 simulations may also be limited in order to stop and examine a specific simulation that leads to convergence problems (see command >>> INCOMPLETE).
If performing Monte Carlo simulations (see command >>> MONTE CARLO), *mtough2* is the number of parameter sets generated and thus the number of realizations of the calculated system output.

Example
```
> COMPUTATION
  >> STOP after
     >>> a total of : 500 TOUGH2 SIMULATIONS
     <<<
  >> ERROR analysis
     >>> MONTE CARLO
     <<<
  <<
```

See Also
>>> ITERATION, >>> LATIN HYPERCUBE, >>> MONTE CARLO, >>> INCOMPLETE

Command
`>>> SINK: sink_name (sink_name_i ...) (+ iplus)`

Parent Command
any second-level command in block `> PARAMETER` or `> OBSERVATION` referring to sink names

Subcommand
any fourth-level command in block `> PARAMETER` or `> OBSERVATION`

Description
(synonym for command `>>> SOURCE`)

Example
(see command `>>> SOURCE`)

See Also
`>>> SOURCE`

Command
```
>> SKIN
```

Parent Command
```
> PARAMETER
```

Subcommand
```
>>> MATERIAL
```

Description
This command selects as a parameter the skin zone radius around a well. In general, the radial thickness of the skin zone is predefined by the discretization of the flow region. In order to estimate the skin zone radius, the TOUGH2 mesh must be generated using the MESHMAKER utility. Furthermore, two zones must be generated using logarithmically increasing radial distances (option LOGAR). The skin radius is defined by the parameter `RLOG` after the first appearance of keyword LOGAR. In addition, an I5 integer must be provided after `NRAD`, `NEQU`, and `NLOG`, indicating the material type number for the corresponding grid blocks (see example below). Estimation of a skin zone radius may be unstable if changing the discretization significantly affects the modeling results.

Example
The following MESHMAKER block is from a TOUGH2 input file. A radial mesh is generated with a borehole of radius 0.1 m (material type 1), a skin zone of material type 2 and initial radius of 0.3 m, partitioned into 20 grid blocks, and an outer zone of material type 3, which extends to a radial distance of 10.0 m. The thickness of the layer is 1.0 m. The skin zone radius to be estimated is printed in italics.

```
MESHMAKER
RZ2D
RADII
    2    1
 0.000E+00 0.100E+00
LOGAR
   20    2 0.300E+00 0.100E-01
LOGAR
   80    3 1.000E+01
LAYER
    1
 0.100E+01
```

The corresponding parameter block in the iTOUGH2 input file reads:

```
> PARAMETER
  >> SKIN zone radius
     >>> MATERIAL: SKINZ (= material name 2 in block ROCKS)
         >>>> ANNOTATION: Skin radius
         >>>> VALUE
         >>>> initial guess: 0.3
         >>>> RANGE        : 0.12  1.00
         <<<<
     <<<
  <<
```

See Also

-

Command
`>>>> SKIP:` *`nskip`*

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
(synonym for command `>>>> HEADER`)

Example
(see command `>>>> HEADER`)

See Also
`>>>> HEADER`

Command
>>> SOURCE: *source_name* (*source_name_i ...*) (+ *iplus*)
or
>>> SINK: *sink_name* (*sink_name_i ...*) (+ *iplus*)

Parent Command
any second-level command in block > PARAMETER or > OBSERVATION referring to source names.

Subcommand
any fourth-level command in block > PARAMETER or > OBSERVATION, respectively

Description
This command reads one or more sink/source code names as specified in the TOUGH2 block GENER. Both parameter and observations may refer to sources.
Sources are designated by a three-character/two-integer code name (TOUGH2 variables *SL* and *NS*; FORTRAN format: A3I2). Blanks in the names as printed in the TOUGH2 output file must be replaced by underscores (e.g., a source name specified in the TOUGH2 input file as 'B 007' is printed as 'B 0 7' in the TOUGH2 output file. Therefore, it must be addressed in the iTOUGH2 input file as 'B_0_7').
Multiple names can be specified. A sequence of *iplus* sources can be generated by increasing the number of the last source code *NS*. The following two command lines are identical:

```
>>> SOURCE: WEL_1  WEL15  +3
>>> SOURCE: WEL_1  WEL15  WEL16  WEL17  WEL18
```

Example
```
> PARAMETER
  >> determine ENTHALPY of injected fluid...
     >>> SOURCE: WEL_1
         >>>> LOGARITHM
         <<<<
     <<<
  << ... based on...

> OBSERVATION
  >> ... the total VAPOR PRODUCTION in the extraction wells
     >>> SINK               : EXT_1 + 3
         >>>> DATA on FILE  : steam.dat
         >>>> RELATIVE error: 10 %
         <<<<
     <<<
```

See Also
-

Command
`>>> STEADY-STATE (SAVE) (: (-)`*`time_step`*`)`

Parent Command
`>> OPTION`

Subcommand
-

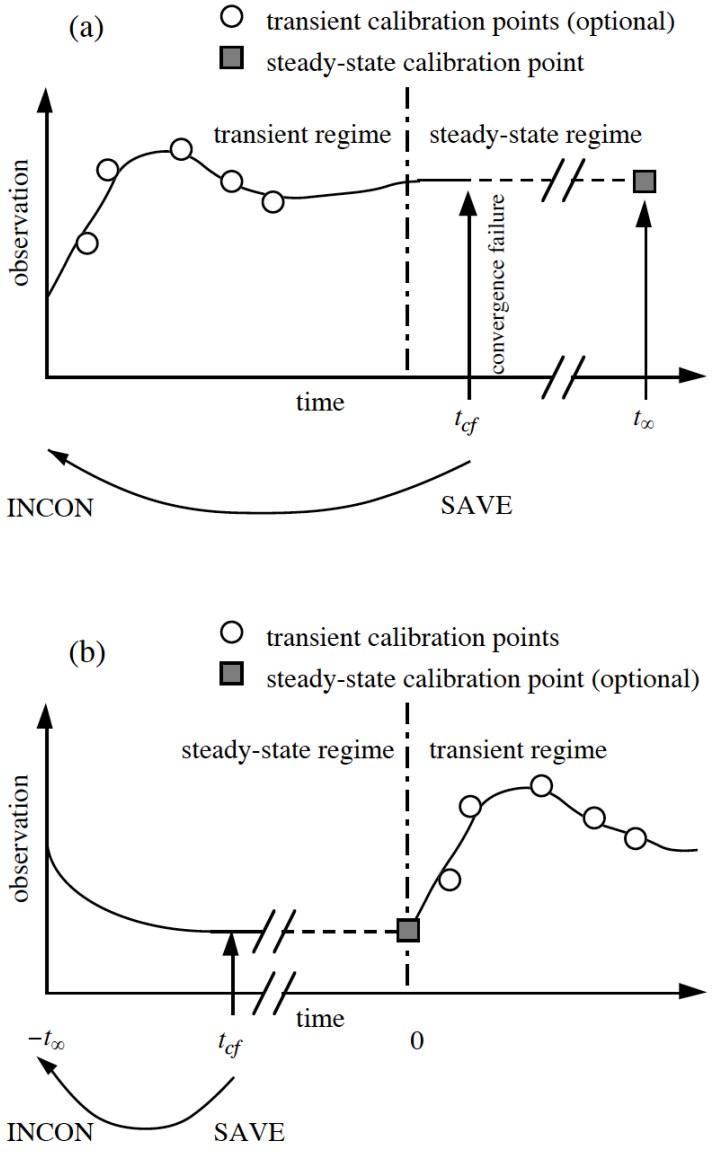
Description
This command allows TOUGH2 to reach steady state prior to or after a transient simulation. This option can be used to calibrate against steady-state data (only or in combination with transient data), or to reach an equilibrium as initial conditions for a subsequent transient simulation.
Note that a transient TOUGH2 simulation must be performed to reach steady state. Steady state is usually indicated by one of the following convergence failures:

- too many time steps converged on a single Newton-Raphson iteration (see command `>>> CONSECUTIVE`);
- convergence failure following two time steps that converged on a single Newton-Raphson iteration;
- too many time step reductions (see command `>>> REDUCTION`);
- maximum number of time steps reached (TOUGH2 variable *`MCYC`*);
- maximum simulation time reached (TOUGH2 variable *`TIMAX`*);
- maximum time step size reached (use a colon on the command line followed by *`time_step`* to select a maximum time step size. The *`time_step`* must be smaller than TOUGH2 variable *`DELTMX`*).
- minimum time step size reached (use a colon on the command line followed by a *negative* *`time_step`* to select a minimum time step size, suggesting convergence problems that are assumed to indicate that steady-state conditions are reached).

iTOUGH2 usually stops if a TOUGH2 run terminates due to one of the above mentioned convergence failures (see command `>>> INCOMPLETE`).
If command `>>> STEADY-STATE` is present, however, it accepts convergence failures and proceeds with the inversion.
Inversions involving steady-state runs can be greatly accelerated by using keyword `SAVE` on the command line. After the first steady-state run has been completed, iTOUGH2 takes the steady-state conditions stored on file *SAVE* as the initial condition (file *INCON*) for the next TOUGH2 run, sets the initial time step to the (usually large) last time step of the previous simulation, and updates the parameter set. Since the parameter set is changed only slightly between subsequent TOUGH2 simulations, the new steady state is usually reached within a

few additional time steps. This option should not be used if also calibrating against transient data preceding the steady-state data point.

Figure (a) below illustrates the use of the steady-state option assuming that calibration against steady-state data is to be performed. A steady-state calibration time t_∞ should be specified at a very late point in time (i.e., beyond the transient phase of the simulation), and the same time should be assigned to the observed steady-state data point in block `>>>> DATA`. (In addition to the steady-state point, one might also specify calibration points during the transient phase). The TOUGH2 simulation proceeds until a convergence failure occurs at an unknown point in time t_{cf}. The primary variables at that point are written to file *SAVE* and reused as initial conditions for the subsequent TOUGH2 run (only if keyword `SAVE` is present). Then, the calculated system state at time t_{cf} is compared to the steady-state data point at time t_∞ (make sure that t_∞ is always greater than t_{cf}, e.g., set $t_\infty = 10^{20}$).



The second application of the steady-state option is illustrated in Figure (b). If a steady-state regime is to precede a transient regime within a single simulation (e.g., to assure initial conditions are at equilibrium, a state that depends on the parameters to be estimated), a negative starting time $-t_\infty$ (TOUGH2 variable *TSTART*) must be specified. Ensure that $|-t_\infty|$ is larger than the duration required to reach steady state, e.g., $-t_\infty = -10^{20}$. The TOUGH2 simulation proceeds until a convergence failure occurs at an unknown point in time t_{cf}, creating the steady-state regime. The primary variables at that point are written to file *SAVE* to be used as initial conditions for the subsequent TOUGH2 simulation (only if keyword `SAVE` is present). The simulation time is then set to zero, and the transient regime of the simulation is initiated. This requires that boundary conditions are changed at time zero, e.g., by starting injection or withdrawal, or by changing Dirichlet-type boundary conditions (see command `>> RESTART TIME`).
(Note that if transient data are combined with steady-state data, the standard deviation of the steady-state data point may have to be decreased substantially to outweigh the large number of transient points.)

Example

```
> OBSERVATION
  >> steady-state point in TIME: 1 (= t_inf)
     1.0E20

  >> PRESSURE
     >>> ELEMENT: FDF76
         >>>> DATA
              1.0E20  1.354E5
         >>>> DEVIATION: 0.05E5
         <<<<
     <<<
  <<

> COMPUTATION
  >> OPTION
     >>> allow STEADY-STATE, use SAVE file for restart
     <<<
  >> CONVERGEnce criteria
     >>> Presume steady-state if : 5 CONSECUTIVE time steps
         converge on ITER=1
     >>> number of ITERATIONS: 5
     <<<
  <<
<
```

See Also

`>> RESTART TIME, >>> CONSECUTIVE, >>> INCOMPLETE, >>> REDUCTION`

Command
```
>>> STEP (UNSCALED): max_step
```

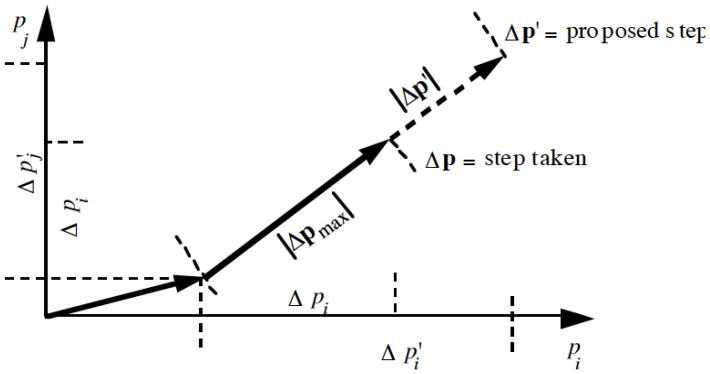
Parent Command
```
>> CONVERGENCE
```

Subcommand
-

Description
This command defines the maximum allowable size of the scaled ($f_i = \sigma_{p_i}$, default; $f_i = p_i$, keyword `CURRENT`) or unscaled ($f_i = 1$, keyword `UNSCALED`) update vector $\Delta\mathbf{p}$ per iTOUGH2 iteration. The step length of the scaled or unscaled parameter update vector is defined as follows:

$$|\Delta\mathbf{p}| = \left[\sum_{i=1}^{n} \left(\frac{\Delta p_i}{f_i} \right)^2 \right]^{1/2}$$

This is a global step size limitation as opposed to the one specified for each individual parameter (see command `>>>> STEP`). Limiting the global step size may make the inversion more stable. Parameter `max_step` should be chosen large enough, so that the size of the proposed step size $|\Delta\mathbf{p}'|$ is reduced to $|\Delta\mathbf{p}_{\max}|$ only during the first few iterations. The figure below illustrates that limiting the step size maintains the direction of the step taken in the parameter space.



Example
```
> COMPUTATION
  >> CONVERGENCE
     >>> number of ITERATIONS     : 10
     >>> limit scaled STEP size to:  1.0 [dimensionless]
     <<<
  <<
```

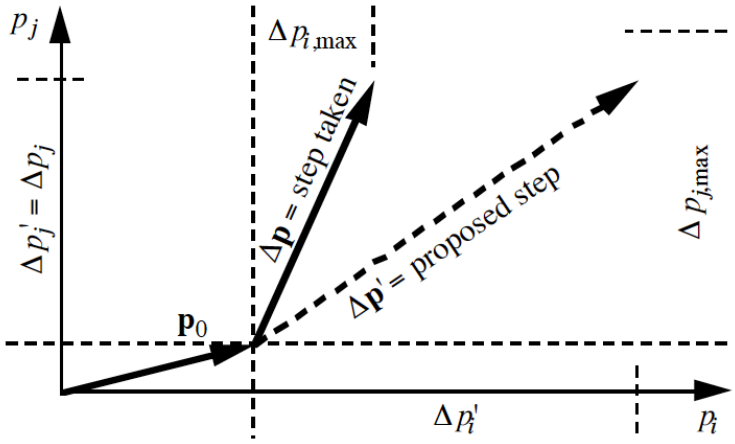
See Also
```
>>>> STEP
```

Command
`>>>> STEP: max_step`

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command defines the maximum allowable step size, limiting the change in the corresponding parameter per iTOUGH2 iteration. Large steps are usually proposed for a parameter if (i) the initial guess is far away from the best estimate, (ii) the parameter has a low sensitivity, or (iii) the parameter is highly correlated to more sensitive parameters. If the system state is a strongly non-linear function of the parameter, even a moderate step size may make the inversion unstable. Parameter `max_step` should be chosen large enough, so that the proposed step size $\Delta p_i'$ is reduced to $\Delta p_{i,\max}$ only during the first few iterations. The figure below illustrates that limiting the step size also changes the direction of the step taken in the parameter space. A global step size limitation can also be specified (see command `>>> STEP`), maintaining the direction of vector $\Delta\mathbf{p}'$.



Example
```
> PARAMETER
  >> POROSITY
     >>> MATERIAL: SANDY
         >>>> standard DEVIATION: 0.10
         >>>> max. STEP size    : 0.05
         <<<<
     <<<
  <<
```

See Also
`>>> STEP, >>>> DEVIATION, >>>> RANGE`

Command
`>>>> STEP: max_step`

Parent Command
`>>> ANNEAL`

Subcommand
-

Description
This command defines an iteration for Simulated Annealing minimization based on the number of successful steps or total number of steps as defined by ***max_step***. After each iteration is completed, the temperature is reduced according to the annealing schedule (see command `>>>> SCHEDULE`). An iteration is completed and the annealing temperature reduced if:

(1) the total number of successful and unsuccessful steps reaches *max_step*, or

(2) (0.2·*max_step*) successful steps have been performed

Example
```
> COMPUTATION
  >> OPTION
     >>> Simulated ANNEALing
         >>>> maximum number of ITERATIONS:  100
         >>>> maximum number of STEPS per ITERATION: 50
         >>>> annealing SCHEDULE: 0.9
         >>>> initial TEMPERATURE: -0.02
         <<<<
     <<<
  <<
```

See Also
`>>>> SCHEDULE`

Command
`>> STOP`

Parent Command
`> COMPUTATION`

Subcommand
(see command `>> CONVERGE`)

Description
(synonym for command `>> CONVERGE`)

Example
(see command `>> CONVERGE`)

See Also
`>> CONVERGE`

Command
```
>>>> SUM
```

Parent Command
any third-level command in block `> OBSERVATION`

Subcommand
-

Description
If multiple elements or connections are provided to indicate the location of a measurement point, this command takes the sum of all calculated values as the model output to be compared to the data. The user must ensure that the summation of the quantity is meaningful.

Example
```
> OBSERVATION
  >> LIQUID FLOW rate
     >>> CONNECTION: A1__1 A2__1 + 7
         >>>> ANNOTATION: Liquid flow across boundary
         >>>> Take SUM of flow rates across 9 connections
         >>>> NO DATA
         <<<<
     <<<
  <<
```

See Also
```
>>>> AVERAGE
```

Command
```
>>> TAU: (-)niter
```

Parent Command
```
>> ERROR
```

Subcommand
-

Description
If multiple observation types (e.g., pressure, flow rate, prior information, etc.) are used to estimate model parameters, the relative weights assigned to each type is given by the ratio $\lambda_{ij} = \tau_i / \tau_j$, where τ_i are scalars such that $\mathbf{C}_i = \tau_i \mathbf{V}_i$. Here, $\mathbf{C}_i$ is the covariance matrix of all observations of type i (a submatrix of covariance matrix $\mathbf{C}_{zz}$), and $\mathbf{V}_i$ is a positive symmetric matrix. By default, τ_i is fixed at 1. This command allows λ_{ij} to be updated in an iterative process, where τ_i is recalculated every multiple of `niter` iterations and is given by:

$$\tau_i = \frac{\mathbf{r}_i^T \mathbf{C}_i^{-1} \mathbf{r}_i}{m_i}$$

This procedure is similar to that sometimes referred to as iterated re-weighted least squares [*Haining*, 1990]. If a negative number is provided for *niter*, prior information is excluded from the process.
Recall that τ_i refers to all data of a certain observation type (i.e., not to individual data sets). The process assigns weights such that the relative contribution of each observation type to the objective function tends to $1/\omega$, where ω is the number of observation types used in the inversion. It should also be realized that the Fisher model test becomes meaningless if λ_{ij} is updated during the inversion because the test will be fulfilled by definition.
While this option provides flexibility in reassigning relative weights to data of different types, it is instead suggested to carefully select the standard deviations of the observations and prior parameter estimates (see command `>>>> DEVIATION (p/o)`) based on potential measurement and modeling errors.

Example
```
> COMPUTATION
  >> ERROR
     >>> update TAU every : -3 rd iteration
     <<<
```

See Also
```
>>>> DEVIATION
```

Command
```
>> TEMPERATURE
```

Parent Command
```
> OBSERVATION
```

Subcommand
```
>>> ELEMENT
```

Description
This command selects temperature as an observation type. This observation type refers to an element.

Example
```
> OBSERVATION
  >> CHANGE in TEMPERATURE
     >>> ELEMENT       : TCP_1
               deg(C)=(deg(F)-32)*0.55
         >>>> SHIFT    : -32.00
         >>>> FACTOR   :   0.55
         >>>> delta(T) DATA in deg(F) on file: Dtemp.dat
         >>>> DEVIATION: 0.2  degree Fahrenheit
         <<<<
     <<<
  <<
```

See Also
-

Command
```
>>>> TEMPERATURE: (-)temp0
```

Parent Command
```
>>> ANNEAL
```

Subcommand
-

Description
This command defines the initial temperature τ_0 for Simulated Annealing minimization. The initial temperature is reduced after each iteration k according to the annealing schedule (see command `>>>> SCHEDULE`). The temperature τ_k defines the probability Π with which an uphill step ΔS should be accepted:

$$\Pi = \exp(-\Delta S / \tau_k)$$

The probability decreases with decreasing temperature. The initial temperature *temp0* should be a reasonable fraction of the objective function S. If a positive value is given, then $\tau_0 =$ *temp0*. If a negative value is provided, the initial temperature is internally calculated as a fraction of the initial objective function:

$$\tau_0 = | \text{temp0} | \cdot S_0$$

Example
```
> COMPUTATION
  >> OPTION
     >>> Simulated ANNEALing
         >>>> maximum number of ITERATIONS:  100
         >>>> maximum number of STEPS per ITERATION: 50
         >>>> annealing SCHEDULE: 0.95
         >>>> initial TEMPERATURE: -0.01 times the initial obj.
func.
         <<<<
     <<<
  <<
```

See Also
```
>>>> STEP (a), >>>> SCHEDULE
```

Command
`>>>> TEMPLATE: num-template-files`

Parent Command
`>>> PEST (c)`

Subcommand
-

Description
iTOUGH2 capabilities can be applied to non-TOUGH2 models using the PEST interface [*Doherty*, 2002]. Template files are used to communicate between iTOUGH2 parameters and the input variables of the user-supplied model, which must be provided through one or multiple ASCII text files. (Note that if the model expects input from the keyboard, the '<' symbol can be used on the command line (see `>>>> EXECUTABLE`) to redirect standard input from the keyboard to a text file.)

A template file must be provided for each input file that contains a parameter adjusted by iTOUGH2; *num-template-files* is the number of template files provided. Template files are matched to their corresponding input files on the lines following the `>>>> TEMPLATE` command. Under Unix, it is recommended to add separate lines with the keyword `FILE:` followed by the file name, so the files are automatically copied to the temporary directory. This can occur anywhere in the iTOUGH2 input file.

Example
```
> COMPUTATION
  >> OPTION
     >>> PEST
         >>>> EXECUTABLE FILE : Run-ModelA-and-ModelB.bat
         >>>> TEMPLATE        : 2
              ModelA.tpl         inputA.txt
              ModelB.tpl         inputB.txt
         >>>> INSTRUCTION     : 1
              outputB.ins        outputB.txt
         <<<<
     <<<

copy FILE: ModelA.tpl  to temporary directory
copy FILE: ModelB.tpl  to temporary directory
copy FILE: outputB.ins to temporary directory
```

See Also
`>>>> EXECUTABLE`, `>>>> INSTRUCTION`

Command
`>> TIME`

Parent Command
`> PARAMETER`

Subcommand
`>>> SOURCE`

Description
This command selects as a parameter the time of a time-dependent generation rate (TOUGH2 variable *F1(L)* for *LTAB* > 1). This parameter refers to a sink/source code name. Index *L* must be provided through command `>>>> INDEX`. Time stepping during simulation should be small compared to the expected variation of the parameter. Furthermore, it is suggested to prescribe the parameter perturbation for calculation of the Jacobian, i.e. use command `>>>> PERTURB` and specify a reasonably large time perturbation provided as a negative number.

Example
```
> PARAMETER
  >> generation TIME
     >>> SOURCE: INJ_1 + 5
         >>>> ANNOTATION: End of spill
         >>>> INDEX     : 2
         >>>> VALUE
         >>>> RANGE     : 1E8  2E8 [sec]
         >>>> PERTURB   :     -1E5 [sec]
         >>>> max. STEP : 86400    [sec]
         <<<<
     <<<
  <<
```

See Also
`>> ENTHALPY, >> RATE`

Command

```
>> TIME: ntime (EQUAL/LOGARITHMIC) (time_unit)
or
>> TIME from DATA/OBSERVATIONS
```

Parent Command

```
> OBSERVATION
```

Subcommand

-

Description

This command selects points in time during a TOUGH2 run at which the calculated system response is used for subsequent iTOUGH2 analysis. In the case of inverse modeling, these are the calibration times at which calculated and observed system responses are compared. If no block `>> TIMES` is specified, each time at which an observation is available will be considered a calibration time. If one (or multiple) lines with command `>> TIMES` are encountered, only the times given in these blocks will be taken as the calibration times, unless command `>> TIMES from DATA` is given, which will add the observation times to the vector of calibration times.

In a given data set, only calibration times within the time window (command `>>>> WINDOW`) are considered; by default, the window is set between the first and last data point of a given data set. Observation times from individual data sets can automatically be added using command `>>>> INDIVIDUAL WINDOWS`.

Time stepping in the TOUGH2 simulation is forced to exactly match all specified calibration times (printout can be suppressed, however, by specifying a negative integer for TOUGH2 parameter *KDATA*). Thus, time stepping considerations must be taken into account when selecting the spacing of calibration points (see also command `>>> CONSECUTIVE`). Only one set of calibration times can be specified which is then applied to all data sets (see, however, command `>>>> (INDIVIDUAL) WINDOW` and `>> TIMES from DATA`).

Values at calibration times that do not match the times at which measurements are available will be linearly interpolated between adjacent data points as illustrated in the figure below.

Calibration times can be specified in three ways: (i) explicitly listed, (ii) automatically generated with equal spacing, and (iii) automatically generated with logarithmic spacing. The corresponding input formats are given by example below. Admissible keywords for selecting the time units *time_unit* are `SECOND`, `MINUTE`, `HOUR`, `DAY`, `WEEK` and `YEAR`.

```
>> provide a list of : 10 TIMES in [MINUTES]
       1.0    2.0   5.0   10.0    15.0   20.0
      30.0   45.0  60.0  120.0   180.0  360.0
```

In the example above, the first *ntime*=10 numbers will be read in free format and interpreted as calibration times in minutes.

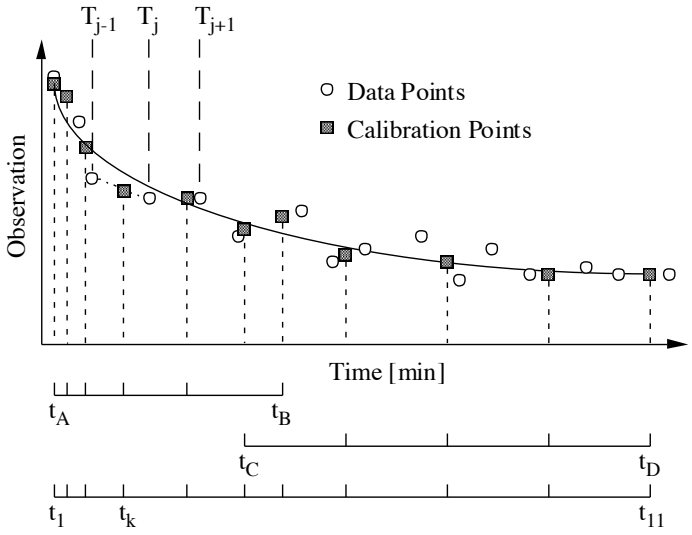
```
>> generate: 10 TIME points, EQUALLY spaced between tA and tB
   60.0  600.0
    tA    tB
```

10 equally spaced points in time will be generated between 60.0 and 600.0 seconds. Time limits *tA* and *tB* are read in free format.

```
>> generate: 10 LOGARITHMICALLY spaced points TIMES in [HOURS]
   0.01  12.0
```

10 logarithmically spaced points in time will be generated between 36 seconds and 12 hours.

The three formats can be used repeatedly, concurrently, and with overlapping time periods (see example below). All specified times will be sorted internally.



Example

The following block generates a total of 11 calibration times; they are schematically shown in the figure above.

```
> OBSERVATION
  >> : 6 LOGARITHMICALLY spaced TIMES in SECONDS between
     30.0   720.0   (tA and tB)

  >> TIME: 5 EQUALLY spaced [MINUTES]
     10.0    30.0   (tC and tD)
```

See Also

>> RESTART TIME, >>> CONSECUTIVE, >>>> WINDOW

Command
`>>> time_unit`
or as a keyword to any command requesting time units

Parent Command
`>> OUTPUT`

Subcommand
-

Description
This command or keyword specifies the time units of all times in the iTOUGH2 output and plot files. Admissible time units are `SECOND` (default), `MINUTE`, `HOUR`, `DAY`, `WEEK`, `MONTH`, `YEAR`, and `DATE`. The time output unit may be different from the time units specified in the input or data files. The `DATE` type supports many different formats, including the following:

November 18, 2003, 09:26:03	18-Nov-2003 09:26:03
18. November 2003, 09:26:03	18.11.2003 09:26:03
18-Nov-2003 09: 26: 03	18/11/2003 9/26/3

The first date supplied to iTOUGH2 is considered the reference date, i.e., time zero for the TOUGH2 simulation

Example
```
> OBSERVATION
  >>  : 7 EQUALLY spaced TIMES in [DAYS]
     1.0 7.0

  >> PRESSURE
     >>> ELEMENT: ZZZ90
         >>>> DATA in [MINUTES] on FILE: pres.dat
         >>>> time WINDOW: 48.0 123.0 [HOURS]
         <<<<
     <<<
  <<

> COMPUTATION
  >> print OUTPUT times
     >>> in WEEKS
     <<<
  <<
```

See Also
-

Command
`>> TOLERANCE`

Parent Command
`> COMPUTATION`

Subcommand
(see command `>> CONVERGE`)

Description
(synonym for command `>> CONVERGE`)

Example
(see command `>> CONVERGE`)

See Also
`>> CONVERGE`

Command
```
>>> TORNADO (: n_std_dev / BOUNDS)
```

Parent Command
```
>> OPTION
```

Subcommand
-

Description
This command generates a Tornado plot by evaluating the objective function at a reference parameter set (i.e., the initial guess) and at discrete points along each of the parameter axes. By default, two additional points per parameter are evaluated by adding and subtracting one standard deviation (as specified by commands `>>>> VARIATION` or `>>>> DEVIATION` in block `> PARAMETER`) from the reference parameter set. Additional points can be requested as `n_std_dev` multiples of the standard deviation. Alternatively, points at the lower and upper `BOUNDS` as specified in command `>>>> RANGE` in block `> PARAMETER` are evaluated.
To obtain a Tornado plot of a particular prediction, use `>>>> ZERO DATA` and `>>> L1-ESTIMATION`.

Example
```
> COMPUTATION
  >> OPTION
     >>> generate TORNADO plot at +/- : 3 standard deviations
     <<<
   <<
```

See Also
`>>> GRID SEARCH, >>> SENSITIVITY`

Command

```
>> TOTAL MASS (phase_name/PHASE: iphase/
               comp_name/COMPONENT: icomp) (CHANGE/DELTA)
```

Parent Command

```
> OBSERVATION
```

Subcommand

```
>>> MATERIAL
>>> MODEL
```

Description

This command selects as an observation type the total mass of phase *iphase* or component *icomp*. This observation type refers either to the entire model domain (command `>>> MODEL`, see also Section A8), or to the subdomain specified by a list of rock types (command `>>> MATERIAL`).

Component number *icomp* or component name *comp_name*, and phase number *iphase* or phase name *phase_name* depend on the EOS module being used. They are listed in the header of the iTOUGH2 output file, and can be specified either on the command line or using the two subcommands `>>>> COMPONENT` and `>>>> PHASE`, respectively. If keyword `CHANGE` is present, the change of the total mass since the first calibration time is computed.

Example

```
> OBSERVATION
  >> CHANGE of TOTAL MASS in place since start of simulation
     >>> entire MODEL
         >>>> ANNOTATION   : Total brine mass injected
         >>>> BRINE is the COMPONENT of interest
         >>>> FACTOR       : 1.0E-03  [g] - [kg]
         >>>> DATA on FILE : brine.dat
         >>>> DEVIATION    :  10.0 [g]
         <<<<
      <<<

  >> TOTAL MASS of PHASE: 2 (= net weight of laboratory column)
     >>> in MATERIAL domain: FRACT MATRI
         >>>> ANNOTATION   : Mass of liquid (water and brine)
         >>>> DATA on FILE : liquid.dat
         >>>> DEVIATION    : 0.01 [kg]
     <<<
   <<
```

See Also

-

Command
>>>> TRIANGULAR

Parent Command
any third-level command in block > PARAMETER

Subcommand
-

Description
This command generates random input parameters following a triangular distribution for Monte Carlo simulations. Parameter values are sampled between the lower and upper bound specified by command >>>> RANGE; the mode is given by command >>>> GUESS.

Example
```
> PARAMETER
  >> RELATIVE permeability function
     >>> DEFAULT
         >>>> PARAMETER         RPD(:  1)
         >>>> ANNOTATION            :  Slr is uncertain
         >>>> sample from TRIANGULAR distribution between...
         >>>> RANGE                 : 0.10 0.50
         >>>> with the initial GUESS: 0.20 as the mode
         <<<<
     <<<
  <<

> COMPUTATION
  >> perform ERROR propagation analysis by means of...
     >>> : 1000 MONTE CARLO simulations using...
     >>> LATIN HYPERCUBE sampling
     <<<
<<
```

See Also
>>> LATIN HYPERCUBE, >>> MONTE CARLO, >>>> GAUSSIAN, >>>> UNIFORM, >>>> NORMAL

Command

```
>>>> TRUNCATE (: (-)trunc)
>>>> EIGENVALUE (: (-)trunc)
>>>> SVD (: (-)trunc)
```

Parent Command

```
>>> SELECT
>>> SUPER
```

Subcommand

-

Description

This command defines one of the criteria used for automatic selection of parameters or superparameters (see *Tonkin and Doherty* [2005]). The parameter space can be truncated based on the eigenvalues of the Fisher Information matrix. Parameter *trunc* defines the cut-off value as the ratio of the eigenvalue to the maximum eigenvalue (default: 10^{-6}). If given as an integer, the top INT(*trunc*) (super)parameters will be selected. If *trunc* is a negative integer, more (super)parameters are added as iterations proceed, with all parameters included for the final iteration.

Example

```
> COMPUTATION
  >> OPTION
     >>> use SUPERparameters automatically every
         >>>> : 3 ITERATIONS
         >>>> TRUNCATE and initially pick top : -3 parameters
         <<<<
      <<<
  <<
```

See Also

```
>>>> ITERATION (s)
```

Command
```
>>>> UNIFORM
```

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command generates uniformly distributed input parameters for Monte Carlo simulations. Parameter values are sampled between the lower and upper bound specified by command `>>>> RANGE`.

Example
```
> PARAMETER
  >> RELATIVE permeability function
     >>> DEFAULT
         >>>> PARAMETER       RPD(:  1)
         >>>> ANNOTATION          :  Slr is uncertain
         >>>> sample from UNIFORM distribution between...
         >>>> RANGE               : 0.01 0.50
         <<<<
     <<<
  <<

> COMPUTATION
  >> perform ERROR propagation analysis by means of...
     >>> :1000 MONTE CARLO simulations using...
     >>> LATIN HYPERCUBE sampling
     <<<
<<
```

See Also
`>>> LATIN HYPERCUBE, >>> MONTE CARLO, >>>> GAUSSIAN, >>>> TRIANGULAR, >>>> NORMAL`

Command
```
>>> UPDATE
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command writes version update information to the iTOUGH2 output file.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> write UPDATE information to iTOUGH2 output file
     <<<
  <<
```

See Also
```
>>> INDEX, >>> VERSION, HELP, LIST
```

Command
>>> UPHILL: *max_uphill*

Parent Command
>> CONVERGE

Subcommand
-

Description
By default, an inversion is stopped if more than 10 consecutive unsuccessful parameter steps have been proposed. The default can be overwritten by *max_uphill*. Each unsuccessful step results in an increase of the Levenberg parameter λ (see command >>> LEVENBERG), leading to a smaller step size deflected towards the gradient.

Example
```
> COMPUTATION
  >> STOP either if
     >>> : 10 ITERATIONS have been completed
     or
     >>> : 3 TOUGH2 runs were INCOMPLETE
     or
     >>> : 5 unsuccessful UPHILL steps have been proposed
     <<<
  <<
```

See Also
>>> ITERATION, >>> LEVENBERG

Command
`>> USER (: anno)`

Parent Command
`> PARAMETER`

Subcommand
```
>>> ELEMENT
>>> MATERIAL
>>> MODEL
>>> SET
>>> SOURCE
```

Description
This command selects a user-specified parameter. This option enables a user to estimate any conceivable parameter, which can be any TOUGH2 variable or function thereof. The user must program the function in subroutine USERPAR, file *it2user.f*. Identification of and details about the parameter can be given in the iTOUGH2 input file and will be transferred to subroutine USERPAR. A parameter annotation *anno* should be provided following a colon on the command line. This annotation (or a substring thereof) will be available in subroutine USERPAR (variable *ANNO*) and can be used to identify the parameter type. The significant part of the string should therefore not be changed by command `>>>> ANNOTATION`. Furthermore, multiple material names as well as grid block or sink/source code names defined by the corresponding third-level command will be transferred to the subroutine in array *NAMEA*. Integers read after command `>>>> INDEX` are provided through array *IDA*. A flag (variable *IVLF*) indicates whether the estimate is a value, logarithm, or multiplication factor of the corresponding parameter.

The user must ensure that all TOUGH2 variables used by the function are transferred to subroutine USERPAR via COMMON blocks. If a variable is not predefined in one of the standard COMMON blocks, a new COMMON block must be created and added to the include file *usercom.inc*.

Subroutine USERPAR has two major blocks. In the first block (*IUIG*=1), the value specified in the TOUGH2 input file is transferred to iTOUGH2 as an initial guess. Programming this first part is optional because initial guesses can also be provided through the iTOUGH2 input file by means of command `>> GUESS`, `>>>> PRIOR`, or `>>>> GUESS`. Programming the second part (*IUIG*=2) is mandatory. In this part, the parameter is updated, i.e., the generic parameter value calculated by iTOUGH2, which is stored in variable *XX*, must be assigned to the appropriate TOUGH2 variable.

The following is the header of subroutine USERPAR in file *it2user.f*, describing the transfer variables. File *it2user.f* must be recompiled before the user-specified parameter becomes active.

```
************************************************************************
      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    = parameter IDs (if needed) (input)                           *
*          the number of elements in IDA is stored in IDA(MAXR-1)      *
* NAMEA  = material or element names (if needed) (input)               *
*          the number of elements in NAMEA is stored in IDA(MAXR)      *
* ANNO   = parameter annotation as specified in iTOUGH2 input file     *
************************************************************************
```

Example

In this simple example, the tortuosity factor (TOUGH2 variable *TORT(NMAT)*) is treated as a user-specified parameter. Variable *TORT* is provided through COMMON block SOLI11 in include file *rock.inc*. The parameter block in the iTOUGH2 input file and subroutine USERPAR are given below.

```
> PARAMETER
  >> USER-specified parameter: TORTUOSITY
     >>> MATERIAL: SAND1
         <<<<
     <<<
  <<
```

```
************************************************************************
      SUBROUTINE USERPAR(IUIG,XX,IVLF,IDA,NAMEA,ANNO)
************************************************************************
C
C$$$$$$$$$ COMMON BLOCKS FOR ROCK PROPERTIES $$$$$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'rock.inc'
C
      CHARACTER ANNO*(*),NAMEA*5
      DIMENSION IDA(*),NAMEA(*)
C
      CALL GETNMAT(NAMEA(1),NMAT)
      IF (IUIG.EQ.1) XX=TORT(NMAT)
      IF (IUIG.EQ.2) TORT(NMAT)=XX
      END
************************************************************************
```

The following is a more general version of subroutine USERPAR for the determination of tortuosity. It allows specifying multiple rock types on command line `>>>> MATERIAL`, and offers the possibility to estimate tortuosity either as a value, a logarithm, or a multiplication factor. Note that identifying the parameter by its annotation is only necessary if other user-specified parameters are also addressed in the same subroutine.

```
************************************************************************
      SUBROUTINE USERPAR(IUIG,XX,IVLF,IDA,NAMEA,ANNO)
************************************************************************
C
C$$$$$$$$$ PARAMETERS FOR SPECIFYING THE MAXIMUM PROBLEM SIZE $$$$$$$$$
      INCLUDE 'maxsize.inc'
C
C$$$$$$$$$ COMMON BLOCKS FOR ROCK PROPERTIES $$$$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'rock.inc'
C
      CHARACTER ANNO*(*),NAMEA*5
      DIMENSION IDA(*),NAMEA(*),X0(MAXROC)
C
C --- Return TOUGH2 value as initial guess
      IF (IUIG.EQ.1) THEN
         IF (ANNO(1:10).EQ.'TORTUOSITY') THEN
            DO 1000 I=1,IDA(MAXR)
               CALL GETNMAT(NAMEA(I),NMAT)
               IF (IVLF.EQ.1) THEN
                  XX=TORT(NMAT)
               ELSE IF (IVLF.EQ.2) THEN
                  XX=LOG10(TORT(NMAT))
               ELSE
                  XX=1.0D0
                  X0(NMAT)=TORT(NMAT)
               ENDIF
 1000       CONTINUE
         ENDIF
C
C --- Update parameter
      ELSE IF (IUIG.EQ.2) THEN
C --- Identify parameter by its annotation
         IF (ANNO(1:10).EQ.'TORTUOSITY') THEN
            DO 1001 I=1,IDA(MAXR)
C --- Obtain material identifier NMAT
               CALL GETNMAT(NAMEA(I),NMAT)
               IF (IVLF.EQ.1) THEN
C --- Estimate is tortuosity value
                  TORT(NMAT)=XX
               ELSE IF (IVLF.EQ.2) THEN
C --- Estimate is logarithm of tortuosity
                  TORT(NMAT)=10.0D0**XX
               ELSE
C --- Estimate is multiplication factor for initial tortuosity
                  TORT(NMAT)=X0(NMAT)*XX
               ENDIF
 1001       CONTINUE
         ENDIF
      ENDIF
      END
************************************************************************
```

Additional examples can be found in file *it2user.f* .

See Also

-

Command
`>> USER (: anno)`

Parent Command
`> OBSERVATION`

Subcommand
`>>> CONNECTION`
`>>> ELEMENT`
`>>> MODEL`
`>>> SOURCE`

Description
This command selects a user-specified observation type. This option enables a user to introduce a new observation type, i.e., any type of data can be used for parameter estimation by defining the corresponding model output, which is a user-specified function of TOUGH2 variables. The user must program the function in subroutine USEROBS, file *it2user.f*. Identification of and details about the observation can be given in the iTOUGH2 input file and will be transferred to subroutine USEROBS. The annotation *anno* (or a substring thereof) will be available in subroutine USEROBS (variable *ANNO*); it can be used to identify the observation type and data set. The significant part of the string should therefore not be changed by subcommand `>>>> ANNOTATION`. Multiple grid block names or sink/source code names defined by the appropriate third-level command will be transferred to the subroutine in array *GRIDA*. The corresponding element numbers are stored in array *NECA*, with *INEC* pointing to the currently processed array index. Integer variables read after command `>>>> INDEX` are provided through array *IOBSA*. On output, subroutine USEROBS returns the user-specified model result *TRESULT*.
The user must ensure that all TOUGH2 variables used by the function are transferred to subroutine USERPAR via COMMON blocks. If a variable is not predefined in one of the standard COMMON blocks, a new COMMON block must be created and added to the include file *usercom.inc*.
The following is the header of subroutine USEROBS, describing the transfer variables. File *it2user.f* must be recompiled before the user-specified observation becomes active.

```
************************************************************************
      SUBROUTINE USEROBS(IUSER,IOBSA,GRIDA,NECA,INEC,ANNO,TRESULT)
************************************************************************
* Provides TOUGH2 result for user-specified observation 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 (input)  *
* INEC   : Current pointer to GRIDA and NECA, respectively (input)     *
* ANNO   : Annotation (input)                                          *
* TRESULT: User-defined TOUGH2 result (output)                         *
************************************************************************
```

Example

In this example the pressure difference measured between two points in a laboratory column is defined as a user-specified observation. Two differential pressures are measured, referring to the liquid and NAPL phase, respectively. The iTOUGH2 input block and subroutine USEROBS are given below. Variable *IOBSA(MAXR-2)* holds the phase number.

```
> OBSERVATION
  >> USER-specified observation type: Pres. Diff.
     >>> difference between ELEMENTs: A1112 A1152
         >>>> ANNOTATION            : Pres. Diff. aqueous phase
         >>>> PHASE                 : 2
         >>>> DATA FILE             : dp_aqu.dat
         >>>> DEVIATION             : 1000.0
         <<<<
     >>> difference between ELEMENTs: A1112 A1152
         >>>> ANNOTATION            : Pres. Diff. NAPL phase
         >>>> NAPL PHASE
         >>>> DATA FILE             : dp_napl.dat
         >>>> DEVIATION             : 1000.0
         <<<<
     <<<
```

```
************************************************************************
      SUBROUTINE USEROBS(IUSER,IOBSA,GRIDA,NECA,INEC,ANNO,TRESULT)
************************************************************************
C$$$$$$$$$ PARAMETERS FOR SPECIFYING THE MAXIMUM PROBLEM SIZE $$$$$$$$$
      INCLUDE 'maxsize.inc'
C$$$$$$$$$ COMMON BLOCKS FOR PRIMARY VARIABLES $$$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'primary.inc'
C$$$$$$$$$ COMMON BLOCK FOR SECONDARY VARIABLES $$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'second.inc'
C --- Identify observation:
      IF (ANNO(1:11).EQ.'Pres. Diff.') THEN
C --- Reference and capillary pressure of first element:
        NEC=NECA(1)
        NLOC=(NEC-1)*NK1
        NLOC2=(NEC-1)*NSEC*NEQ1+(IOBSA(MAXR-2)-1)*NBK
        PREF1=X(NLOC+1)
        PCAP1=PAR(NLOC2+6)
        P1=PREF1+PCAP1
C --- Reference and capillary pressure of second element:
        NEC=NECA(2)
        NLOC=(NEC-1)*NK1
        NLOC2=(NEC-1)*NSEC*NEQ1+(IOBSA(MAXR-2)-1)*NBK
        PREF2=X(NLOC+1)
        PCAP2=PAR(NLOC2+6)
        P2=PREF2+PCAP2
C --- Take the difference
        TRESULT=P2-P1
      ENDIF
      END
```

See Also

-

Command
>>>> USER

Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
Observed data can be represented as a user-specified function of time. The function must be coded into subroutine USERDATA, file *it2user.f*. The following is the header of subroutine USERDATA, describing the transfer variables. File *it2user.f* must be recompiled before the user-specified data definition becomes active.

```
************************************************************************
      SUBROUTINE USERDATA(IDF,TIME,ANNO,F)
************************************************************************
* User specified function to represent observed data                   *
* IDF : data set identifier (input)                                    *
* TIME: time at which data are to be provided (input)                  *
* ANNO: annotation (input)                                             *
* F   : value of observed data at time TIME (output)                   *
************************************************************************
```

Example
In this example an analytical solution is provided as data to be matched (see sample problem 8). Liquid saturation in response to a liquid pulse is calculated as a function of time and location in subroutine USERDATA (see following page). Note that the Z-coordinate is passed to USERDATA through the annotation.

```
> OBSERVATIONS

  >> LIQUID SATURATION
     >>> GRID BLOCK      : BP1_1
         >>>> ANNOTATION : ANALYT. Z=0.100
         >>>> DEVIATION  : 0.002
         >>>> analytical solution in USER-specified function
         <<<<

     >>> GRID BLOCK      : BZ1_1
         >>>> ANNOTATION : ANALYT. Z=0.200
         >>>> DEVIATION  : 0.002
         >>>> analytical solution in USER-specified function
         <<<<
     <<<
```

```
************************************************************************
      SUBROUTINE USERDATA(IDF,TIME,ANNO,F)
************************************************************************
* User specified function to represent observed data                   *
* IDF : data set identifier (input)                                    *
* TIME: time at which data are to be provided (input)                  *
* ANNO: annotation (input)                                             *
* F   : value of observed data at time TIME (output)                   *
************************************************************************
C
C$$$$$$$$$ PARAMETERS FOR SPECIFYING THE MAXIMUM PROBLEM SIZE $$$$$$$$$$
      INCLUDE 'maxsize.inc'
C
C$$$$$$$$$ COMMON BLOCKS FOR SIMULATION PARAMETERS $$$$$$$$$$$$$$$$$$$$
      INCLUDE 'param.inc'
C
C$$$$$$$$$ COMMON BLOCKS FOR ROCK PROPERTIES $$$$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'rock.inc'
C
C$$$$$$$$$ COMMON BLOCK FOR SECONDARY VARIABLES $$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'second.inc'
C
C$$$$$$$$$ COMMON BLOCKS FOR TOTAL MASS AND VOLUMES $$$$$$$$$$$$$$$$$$
      INCLUDE 'rmasvol.inc'
C
C$$$$$$$$$ COMMON BLOCKS FOR ELEMENTS $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      INCLUDE 'elements.inc'

      CHARACTER ANNO*(*)
      SAVE A

      F=0.0D0
C --- Analytical solution 1D-pulse
      IF (ICALL.EQ.1) A=CP(1,1)
      IF (ANNO(1:10).EQ.'ANALYT. Z=') THEN
         READ(ANNO(11:15),'(F5.3)',IOSTAT=IOS) Z
         V=PER(3,1)*PAR(4)*GF/POR(1)/PAR(3)
         DIFF=A*PER(3,1)/POR(1)/PAR(3)
         XM=XPVOLU0(2)/POR(1)
         F=XM/(DSQRT(4.0D0*3.1416D0*DIFF*TIME))*
     &     DEXP(-(Z-V*TIME)**2/(4.0D0*DIFF*TIME))
      ENDIF
      END
C --- END of USERDATA
```

See Also

>>>> DATA, >>>> POLYNOM

Command
```
>>>> VALUE
```

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command selects the estimation of the parameter value as opposed to its logarithm (see command `>>>> LOGARITHM (p)`) or a multiplication factor (commands `>>>> FACTOR (p)` or `>>>> LOG(F)`).

Example
```
> PARAMETER
  >> parameter of RELATIVE permeability function
     >>> MATERIAL: SAND1
         >>>> estimate VALUE of ...
         >>>> ANNOTATION         :  Resid. Liq. Sat.
         >>>> PARAMETER no.      :   1
         <<<<
     <<<
  <<
```

See Also
`>>>> FACTOR (p), >>>> LOGARITHM (p), >>>> LOG(F)`

Command
>>>> VARIANCE: *sigma^2* (ADD NOISE)

Parent Command
any third-level command in block > PARAMETER and > OBSERVATION

Subcommand
-

Description
(see >>>> DEVIATION (p/o))

Example
(see >>>> DEVIATION (p/o))

See Also
>>>> DEVIATION (p/o)

Command
`>>>> VARIATION:` *sigma*

Parent Command
any third-level command in block `> PARAMETER`

Subcommand
-

Description
This command specifies the expected variation σ_{p_j} of a parameter. σ_{p_j} is used to scale the columns of the Jacobian matrix, yielding dimensionless and comparable sensitivity coefficients. While the solution of the inverse problem is not affected by the choice of the scaling factor, all the qualitative sensitivity measures are directly proportional to σ_{p_j}:

$$\text{scaled sensitivity coefficient: } S_{ij} = \frac{\partial z_i}{\partial p_j} \cdot \frac{\sigma_{p_j}}{\sigma_{z_i}} = J_{ij} \cdot \frac{\sigma_{p_j}}{\sigma_{z_i}}$$

If no standard deviation (see `>>>> DEVIATION (p)`) or parameter variation is specified, the scaling factor is taken to be 10 % of the respective parameter value.
For sensitivity analyses, σ_{p_j} can be taken as the perturbation one would apply to study the effect of the parameter on the modeling result. As `>>>> VARIATION` indicates that no prior information is available, the data-worth analysis will examine the value of adding such information.

Example
```
> PARAMETER
  >> POROSITY
     >>> MATERIAL: FAULT
         >>>> expected VARIATION : 0.10
         <<<<
     <<<
  >> ABSOLUTE permeability
     >>> MATERIAL: FAULT
         >>>> expected VARIATION : 1.0 (one order of magnitude)
         <<<<
     <<<
  <<
> COMPUTATION
  >> OPTION
     >>> SENSITIVITY analysis
     <<<
```
See Also
`>>>> DEVIATION (p), >>> WORTH (op)`

Command
```
>>> VERSION
```

Parent Command
```
>> OUTPUT
```

Subcommand
-

Description
This command prints version control statements at the end of the iTOUGH2 output file. The same information is always written to the **.msg* file. If making code modifications it is strongly suggested to update the version control statement at the beginning of each subroutine or function.

Example
```
> COMPUTATION
  >> OUTPUT
     >>> VERSION control statements
     <<<
```

On output:

```
-------------------------------------------------------------------------------------
 PROGRAM  VERSION   DATE             COMMENT
-------------------------------------------------------------------------------------
 iTOUGH2            Current version  iTOUGH2 V6.9 (JULY, 2014)
 WHATCOM  1.0      10 AUGUST    1993 #35: Q: WHAT COMPUTER IS USED? A: LINUX
 CALLSIG  1.0       5 DECEMBER  1995 #112: SIGNAL HANDLER
 CPUSEC   1.0      10 AUGUST    1993 #--: RETURNS CPU-TIME (VERSION LINUX)
 OPENFILE 4.0      19 JANUARY   1999 #31: OPENS MOST OF THE FILES
 LENOS    1.0       1 MARCH     1992 #28: RETURNS LENGTH OF LINE
 PREC     1.0       1 AUGUST    1992 #86: CALCULATE MACHINE DEPENDENT CONSTANTS
 ITHEADER 3.2      27 MAY       1998 #29: PRINTS iTOUGH2 HEADER
 DAYTIM   1.0      10 AUGUST    1993 #32: RETURNS DATE AND TIME (VERSION SUN)
 THEADER  5.0      12 JULY      2002 #30: PRINTS TOUGH2 HEADER
 SOLVTYPE 1.0       1 OCTOBER   1999 INITIALIZE PARAMETERS FOR THE SOLVER PACKAGE
 ......   ...      .. ........  .... ................................................
```

See Also
```
>>> UPDATE
```

Command
>> VOLUME (*phase_name*/PHASE: *iphase*) (CHANGE/DELTA)

Parent Command
> OBSERVATION

Subcommand
>>> MATERIAL
>>> MODEL

Description
This command selects as an observation type the total volume of phase *iphase*.
This observation type refers either to the entire model domain (command >>> MODEL), or to the subdomain specified by a list of rock types (command >>> MATERIAL).
The phase name *phase_name* or phase number *iphase*, which depend on the EOS module being used, are listed in the iTOUGH2 header. They can be specified either on the command line or using the subcommand >>>> PHASE.
If keyword CHANGE is present, the change of the total volume since the first calibration time is computed. This option can be used, for example, to calculate the cumulative volumetric liquid flow, which is equivalent to the change of the total gas volume in place.

Example
```
> OBSERVATION
  >> CHANGE in total GAS VOLUME
     >>> entire MODEL
         >>>> ANNOTATION: Cum. vol. water flux
         >>>> FACTOR    : 1.0E-06  [ml] - [m^3]
         >>>> DATA [DAYS]
              0.001   0.00
              0.10  245.16
              0.25  354.84
              0.50  419.35
              1.00  470.97
              2.10  522.58
              2.50  539.35
              3.00  552.26
         >>>> DEVIATION :  5.0 [ml]
         <<<<
      <<<
   <<
```

See Also
-

Command
```
>>> WARNING
```

Parent Command
```
>> CONVERGE
```

Subcommand
-

Description
iTOUGH2 checks the consistency of the TOUGH2 and iTOUGH2 input, printing error and warning messages to the iTOUGH2 output file. The program stops if an error or warning is encountered. Command `>>> WARNING` makes iTOUGH2 continue despite the occurrence of a warning message.
It is strongly suggested to resolve all warning messages before continuing. Warning messages should only be ignored if the reason for message is completely understood and deemed harmless.

Example
```
> COMPUTATION
  >> STOP
     >>> don't stop because of WARNINGS
     <<<
  <<
```

See Also
```
>>> INPUT
```

Command
>> WATERTABLE

Parent Command
> OBSERVATION

Subcommand
>>> ELEMENT

Description
This command selects as an observation type the elevation of the watertable in the coordinate system given in Columns 71–80 of the TOUGH2 ELEME block. Multiple elements must be provided (preferably a vertical column, e.g., representing a well), covering the expected range of watertable fluctuations. A warning message will be printed if the entire column is either (1) fully saturated (i.e., the water table is above the highest specified element), (2) unsaturated (i.e., the water table is below the lowest specified element), or (3) has a perched water body within the specified interval. Generally, the watertable elevation is assumed to be within the lowest-elevation element that has a saturation below the residual gas saturation (i.e., only trapped gas). The calculation of the water table is based on the assumption that Z-coordinate in the TOUGH2 ELEME block refers to the center of the element, and that nodal distances are proportional to element volumes. The option also requires that a gravitational accellearation is given (variable GF in the TOUGH2 block PARAM).

Example
```
> OBSERVATION
  >> WATERTABLE elevation
     >>> all ELEMENTS in COORDINATE BOX: 0. 1. -25.  0. 1. -20.
               This coordinate box covers the expected range of
               watertable fluctuations (-25 to -20 m) in Well 5,
               which is located at X=0.5 and Y=0.5
         >>>> ANNOTATION: WT in Well 5
         >>>> SHIFT     : -95.0 m (reference datum at 95 masl)
         >>>> DATA in [HOURS] from FILE: WT_Well5.dat
         >>>> DEVIATION :  0.1 m
         <<<<
      <<<
   <<
```

See Also
>> DRAWDOWN

Command
`>>>> WEIGHT:` *`1/sigma`* `(ADD NOISE)`

Parent Command
any third-level command in block `> PARAMETER` and `> OBSERVATION`

Subcommand
-

Description
(see `>>>> DEVIATION (p/o)`)

Example
(see `>>>> DEVIATION (p/o)`)

See Also
`>>>> DEVIATION (p/o)`

Command
>>>> WINDOW (INDIVIDUAL / : *time_A time_B* (*time_unit*))

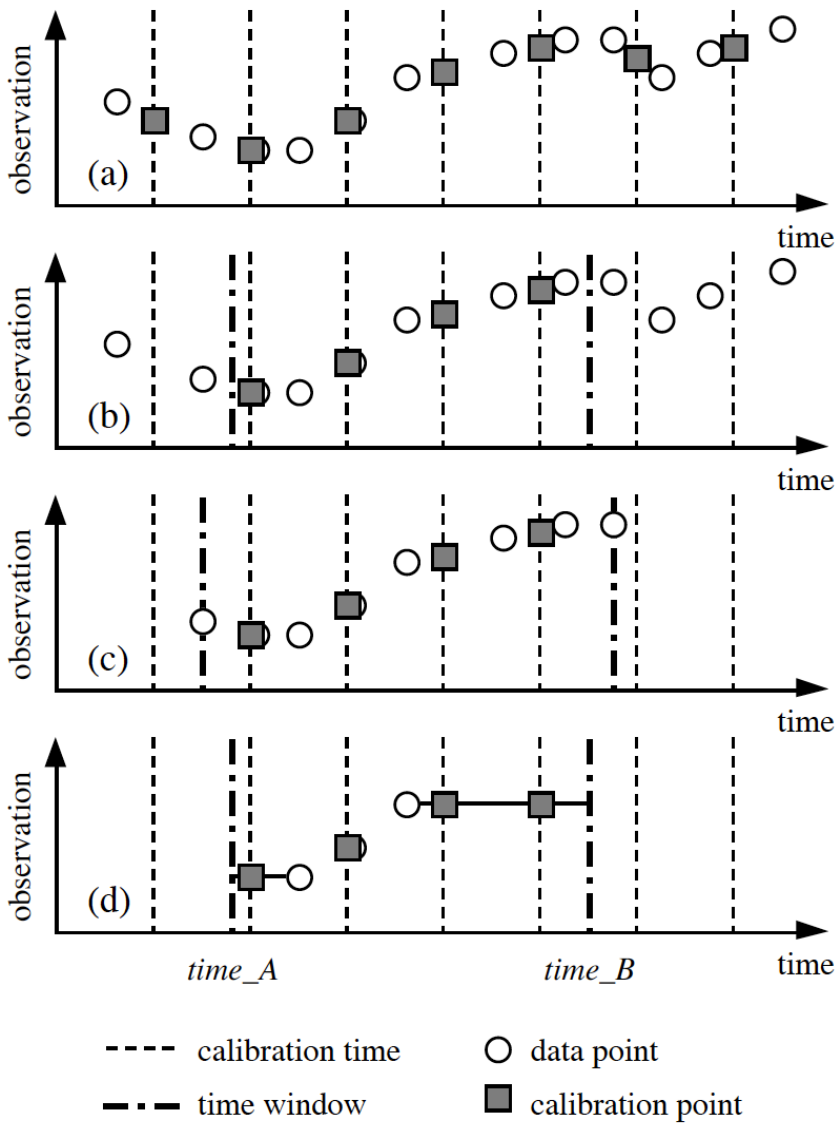
Parent Command
any third-level command in block > OBSERVATION

Subcommand
-

Description
The times at which the observed and calculated system responses are compared are referred to as calibration times. They are specified using command >> TIME. The calibration times are defined globally, i.e., there is only one set of calibration times and it is applied to all data sets. Calibration times and observation times may be different; observed data are linearly interpolated between adjacent data points to obtain a value at the calibration time.
In the standard case, the first data point is observed at or before the first calibration time, and the last data point is observed at or after the last calibration time. This configuration ensures that a value for calibration is available or can be interpolated at every calibration point. The standard situation is sketched in Figure (a) below, assuming that equally spaced calibration times (dashed lines) have been defined, i.e., using the following command line:

```
>> TIMES: 7 EQUALLY SPACED
```

If, for some reason, only a subset of the available data can be used for calibration as shown in Figure (b) below, the user must specify the time window between which data should be processed: *time_A* and *time_B* are the lower and upper bounds of the window. Only one time window may be specified per data set. If more than one time window is required, new data sets containing the same data must be generated and different time windows specified for each data set. The time units of *time_A* and *time_B* can be specified using one of the *time_unit* keywords.
If a data set does not contain data that extend over the entire range of calibration times, iTOUGH2 automatically adjusts the time window to coincide with the first and last data point as shown in Figure (c). A warning message is printed, however, which can be avoided by explicitly specify the time window.
If the specified time window is larger than the available data set, the value of the first and last data point is horizontally extrapolated to *time_A* and *time_B*, respectively, as indicated in Figure (d).
If keyword INDIVIDUAL is present, individual time windows will be set around each data point, and the observation times will be automatically added to the vector of calibration times. No *time_A* and *time_B* need to be specified in this case.
If time is shifted, the time window must be given with reference to the time system of the data, if command >>>> SHIFT TIME: *tshift* appears before command >>>> WINDOW. Conversely, the window must be given in the shifted time system, if the shift command appears after command >>>> WINDOW.



Example

```
> OBSERVATION
  >> :7 EQUALLY spaced TIMES in [MINUTES]
      5.0  35.0
  >> PRESSURE
     >>> ELEMENT: ZZZ99
         >>>> DATA [HOURS] on FILE: pres.dat
         >>>> time WINDOW: 299.0  901.0 [SECONDS]
         >>>> DEVIATION  : 1.E3
         <<<<
     <<<
```

See Also

```
>> TIME, >>>> SHIFT
```

Command

```
>>> WORTH (PARAMETER/PREDICTION) (DETERMINANT/TRACE) (SET)
          (METRIC: imetric) (PERCENT)
```

Parent Command

```
>> OPTION
```

Subcommand

-

Description

This command makes iTOUGH2 perform a data-worth analysis. The significance of an observation for an inversion (or prediction) is assessed by evaluating how much the parameter uncertainty (keyword `PARAMETER`) or prediction uncertainty (keyword `PREDICTION`) is increased when adding a potential (see command `>>>> POTENTIAL`) or removing an existing calibration data point or data set. This impact is measured either by the trace (keyword `TRACE`; default) or determinant (keyword `DETERMINANT`) of the respective covariance matrix. If keyword `SET` is present, the data-worth analysis is performed for entire data sets, i.e., not individual data points. The value of adding or removing prior information about parameters is also examined (see command `>>>> VARIATION`).

If one or multiple predictions are specified (see command `>>>> PREDICTION`), the data-worth analysis is performed based on the trace of the prediction covariance matrix; if no such predictions are specified, the data-worth analysis can only be based on either the trace or the determinant of the estimation covariance matrix.

The data-worth analysis is performed whenever the Jacobian matrix and estimation or prediction covariance matrices are evaluated (i.e., also for a sensitivity analysis and derivative-based optimization). In these cases, use command `>>> WORTH` as a subcommand of `>> OUTPUT` to select printout options.

The data-worth analysis requires n+1 or 2n+1 simulation runs to calculate the Jacobian matrix using forward or centered finite differences, respectively. The data-worth analysis can be computationally costly if the significance of many potential observations is evaluated (note, however, that the number of predictions has essentially no effect on computation cost).

The metric used to evaluate data worth can be selected using keyword `METRIC`, followed by an integer that identifies the function (see separate manual).

The data worth can be printed directly or as a percentage (keyword `PERCENT`) of the total data worth of all actual or potential calibration points. Background information about data-worth analysis can be found in *Dausman et al*. [2010] and *Wainwright and Finsterle* [2015].

Example

```
> COMPUTATION
  >> OPTION
     >>> evaluate WORTH of entire data SETs based on TRACE
```

See Also

```
>>>> POTENTIAL, >>>> PREDICTION, >>>> VARIATION, >>> WORTH (ou)
```

Command

```
>>> WORTH (PARAMETER/PREDICTION) (DETERMINANT/TRACE) (SET)
          (METRIC: imetric) (PERCENT)
```

Parent Command

```
>> OUTPUT
```

Subcommand

-

Description

The data-worth analysis is available whenever the Jacobian matrix and estimation or prediction covariance matrices are evaluated (i.e., for sensitivity analysis and derivative-based optimization). However, since the analysis may be costly if many observations or data sets are involved, it is only executed if the number of observations is less than 500, or if the explicit purpose of the iTOUGH2 run is to perform a data-worth analysis (i.e., when command `>>> WORTH (op)` is specified).

This command forces iTOUGH2 to perform a data-worth analysis even if there are more than 500 observations or data sets. It also allows the user to select the basis for the analysis. For details about the data-worth analysis, see command `>>> WORTH (op)`.

If keyword `SET` is present, the data-worth analysis is performed for entire data sets, i.e., not individual data points.

The metric used to evaluate data worth can be selected using keyword `METRIC`, followed by an integer that identifies the function (see separate manual).

The data worth can be printed directly or as a percentage (keyword `PERCENT`) of the total data worth of all actual or potential calibration points. Background information about data-worth analysis can be found in *Dausman et al*. [2010] and *Wainwright and Finsterle* [2015].

Example

```
> COMPUTATION
  >> OUTPUT
     >>> print data-WORTH analysis results based on  &
         DETERMINANT of PREDICTION covariance matrix &
         using METRIC: 1
     <<<
```

See Also

```
>>> WORTH (op)
```

ACKNOWLEDGMENT

iTOUGH2 (up to version 1.1) was developed at the Laboratory of Hydraulics, Hydrology, and Glaciology (VAW) of the Swiss Federal Institute of Technology (ETH), Zürich, Switzerland, in collaboration with the Swiss National Cooperative for the Disposal of Radioactive Waste (Nagra), Wettingen, Switzerland. Subsequent versions were supported, in part, by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Geothermal Technologies, of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098, and by a grant from the Swiss National Cooperative for the Disposal of Radioactive Waste (Nagra), Wettingen, Switzerland.

Many people have made valuable suggestions and contributions during the long and on-going development of iTOUGH2. I would like to thank Jürg Trösch, Uli Kuhlmann (VAW) and Stratis Vomvoris (Nagra) for their support during the early stages of the project. Even though the main avenues of iTOUGH2 applications have been used extensively, there are many combinations of options that remain to be tested. Many errors, flaws, and inconsistencies have been detected by those who tried to apply earlier versions of iTOUGH2, and many new capabilities have been proposed. iTOUGH2 has greatly benefited from this phase of beta-testing. I would like to acknowledge the contributions of Rainer Senger (INTERA), Steve Webb (Sandia National Laboratories, Albuquerque), Steve White (Industrial Research Ltd., New Zealand), E. Roeder (Clemson University), Torben Sonnenborg (Technical University of Denmark), Nils-Otto Kitterød (University of Oslo, Norway), Alfredo Battistelli (Aquater S.p.A., Italy), Grimur Björnsson (Orkustofnun, Reykjavik, Iceland), Magnus Jonsson (University of Iceland), Tsuyoshi Hiratsuka and Mikio Takeda (AIST, Japan) as well as Chris Doughty, Rick Ahlers, Hui-Hai Liu, Teamrat Ghezzehei, Yingqi Zhang, Mike Kowalsky, Yu-Shu Wu, Frank Hale, Haruko Wainwright, Lilja Magnusdottir, Yoojin Jung, Joseph Doetsch, George Pau, Jihoon Kim, and John Edmiston (LBNL). Special thanks are due to David Bullivant and Mike O'Sullivan (University of Auckland, New Zealand) who helped me restructure and streamline the coding of iTOUGH2, and to April James and Chris Doughty (LBNL) who tested some of the less frequently used options, and who thoroughly reviewed Chapter 4 of this report, making many of the command descriptions less cryptic. Ron Falta (Clemson University) gave me the opportunity to write an adaptation of iTOUGH2 for PCs. Chris Doughty and Ardyth Simmons (LBNL) are thanked for their careful reviews of an earlier draft of this report.

If iTOUGH2 turns out to be a useful tool, this is mainly due to its powerful engine, the simulator TOUGH2. I thank Karsten Pruess for his continuous support.

NOMENCLATURE

α	level of significance; $(1-\alpha)$ is confidence level
$\mathbf{C}_{pp}$	$n \times n$ covariance matrix of estimated parameters; $\mathbf{C}_{pp} = s_0^2\left(\mathbf{J}^T\mathbf{V}_{zz}^{-1}\mathbf{J}\right)^{-1}$
$\mathbf{C}_{zz}$	$m \times m$ *a priori* covariance matrix of measurement error,
$\mathbf{C}_{\hat{z}\hat{z}}$	$m \times m$ *a posteriori* covariance matrix of predicted system response, $\mathbf{C}_{\hat{z}\hat{z}} = \mathbf{J}\mathbf{C}_{pp}\mathbf{J}^T$
$\mathbf{J}$	$m \times n$ Jacobian matrix with elements $J_{ij} = \partial z_i / \partial p_j$
λ	Levenberg parameter
m	number of calibration points
n	number of parameters
ν	Marquardt parameter
$\mathbf{p}$	parameter vector (dimension n)
$\mathbf{p}^*$	prior information vector (dimension n)
$\mathbf{p}_0$	vector of initial parameter guesses (dimension n)
$\mathbf{r}$	residual vector (dimension m) with elements $r_i = z_i^* - z_i(\mathbf{p})$
σ_i^2	*a priori* error variance
s_0^2	*a posteriori* or estimated error variance; $s_0^2 = \dfrac{\mathbf{r}^T\mathbf{V}_{zz}^{-1}\mathbf{r}}{m-n}$
S	objective function, e.g., sum of squared weighted residuals
t	time
$\mathbf{V}_{zz}^{-1}$	$m \times m$ weighting matrix, where $\mathbf{C}_{zz} = \sigma_0^2 \cdot \mathbf{V}_{zz}$
$\mathbf{x}$	coordinate vector (dimension 3)
X	TOUGH2 input parameter
X_0	initial guess of TOUGH2 input parameter
$\mathbf{z}$	vector of observable variables (dimension m)
$\mathbf{z}^*$	measurement vector (dimension m), including prior information
$\hat{\mathbf{z}}$	predicted system response (dimension m)

REFERENCES

Box, G, and Cox, D., An analysis of transformations, *J. Royal Statistical Society, Series B, 26*(2), 211–252, 1964.

Carslaw, H.S. and J.C. Jaeger, 1959. *Conduction of Heat in Solids, 2nd edition*. Oxford University Press, Oxford, pp. 336

Dausman, A.M., J. Doherty, C.D. Langevin, and M.C. Sukop, Quantifying data worth toward reducing predictive uncertainty, *Ground Water*, *48*(5), 729–740, 2010.

Deutsch, C.V., and A.G. Journel, *GSLIB, Geostatistical Software Library and User's Guide*, Oxford University Press, New York, New York, 1992.

Doherty, J., *PEST: Model-Independent Parameter Estimation*, Watermark Numerical Computing, Australia, 2002.

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

Doughty, C. A., *Estimation of Hydrologic Parameters of Heterogeneous Geologic Media with an Inverse Method Based on Iterated Function Systems*, Report LBL-38136, Lawrence Berkeley Laboratory, Berkeley, Calif., 1995.

Falta, R. W., K. Pruess, S. Finsterle, and A. Battistelli, *T2VOC User's Guide*, Report LBL-36400, Lawrence Berkeley Laboratory, Berkeley, Calif., 1995.

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

Finsterle, S., *iTOUGH2 User's Guide,* Report LBNL-40401, Lawrence Berkeley National Laboratory, Berkeley, Calif., 2007a.

Finsterle, S., *iTOUGH2 Command Reference,* Report LBNL-40401 (Updated reprint), Lawrence Berkeley National Laboratory, Berkeley, Calif., 2007b.

Finsterle, S., *iTOUGH2 Sample Problems*, Report LBNL-40402 (Updated reprint), Lawrence Berkeley National Laboratory, Berkeley, Calif., 2007c.

Finsterle, S., G. J. Moridis, and K. Pruess, *A TOUGH2 Equation-of-State Module for the Simulation of Two-Phase Flow of Air, Water, and a Miscible Gelling Liquid*, Report LBL-36086, Lawrence Berkeley Laboratory, Berkeley, Calif., 1994.

Finsterle, S., and M.B. Kowalsky, *iTOUGH2-GSLIB User's Guide*, Report LBNL/PUB-3191, Lawrence Berkeley National Laboratory, Berkeley, Calif., 2007.

Finsterle, S., *Enhancements to the TOUGH2 Simulator Implemented in iTOUGH2*, Report LBNL-TBD, Lawrence Berkeley National Laboratory, Berkeley, Calif., 2015.

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

Haining, R., *Spatial Data Analysis in the Social and Environmental Sciences*, Cambridge University Press, Cambridge, 1990.

International Formulation Committee of the 6th International Conference on the Properties of Steam: The 1967 IFC Formulation for Industrial Use, Verein Deutscher Ingenieure, Düsseldorf, Germany, 1967.

International Association for the Properties of Water and Steam: Revised Release on the IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use, IAPWS Release, The Netherlands, 1995.

International Association for the Properties of Water and Steam: Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam, IAPWS Release, Switzerland, 2007.

Kitterød, N.-O., and L. Gottschalk, Simulation of normal distributed smooth fields by Karhunen-LoŽve expansion in combination with kriging, *Stochastic Hydrology and Hydraulics*, *11*, 459–482, 1997.

Levenberg, K., A method for the solution of certain nonlinear problems in least squares, *Quart. Appl. Math., 2*, 164–168, 1944.

Liu, H.-H., C. Doughty, and G. S. Bodvarsson, An active fracture model for unsaturated flow and transport in fractured rocks, *Water Resour. Res.*, *34*(10), 2633–2646, 1998.

Luckner, L., M. Th. van Genuchten, and D. Nielsen, A consistent set of parametric models for the two-phase flow of immiscible fluids in the subsurface, *Water Resour. Res.*, *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.

Moridis, G. J., and K. Pruess, *Flow and Transport Simulations Using T2CG1, a Package of Conjugate Gradient Solvers for the TOUGH2 Family of Codes*, Report LBL-36235, Lawrence Berkeley Laboratory, Berkeley, Calif., 1995.

Morris, M.D., Factorial sampling plans for preliminary computational experiments, *Technometrics*, *33*(2), 161–174, 1991.

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

Pruess, K., *TOUGH2—A General-Purpose Numerical Simulator for Multiphase Fluid and Heat Flow*, Report LBL-29400, Lawrence Berkeley Laboratory, Berkeley, Calif., 1991a.

Pruess, K., *EOS7, an Equation-of-State Module for the TOUGH2 Simulator for Two-Phase Flow of Saline Water and Air*, Report LBL-31114, Lawrence Berkeley Laboratory, Berkeley, Calif., 1991b.

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

Skahill, B.E., and J. Doherty, Efficient accommodation of local minima in watershed model calibration, *J. Hydrol.*, *329*, 122–139, doi:10.1016/j.jhydrol.2006.02.005, 2006.

Tonkin, M. J., and J. Doherty, A hybrid regularized inversion methodology for highly parameterized environmental models, *Water Resour. Res.*, *41*, W10412, doi:10.1029/2005WR003995, 2005.

Wainwright, H.M., and S. Finsterle, *Global Sensitivity and Data-Worth Analysis in iTOUGH2*, *User's Guide*, Report LBNL-TBD, Lawrence Berkeley National Laboratory, Berkeley, Calif., 2015.

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

Zhang, Y., L. Pan, K. Pruess, and S. Finsterle, A time-convolution approach for modeling heat exchange between a wellbore and surrounding formation, *Geothermics*, *40*(4), 251–266, doi:10.1016/j.geothermics.2011.08.003, 2011.

APPENDIX A: Command Index

```
>> POISSON
>> POROSITY
>> PRODUCTIVITY INDEX
>> PUMPING RATIO
>> RATE
>> REFLECTION
>> REGION (SINK/SOURCE, PERMEABILITY, OBSERVATION)
>> REGRESSION
>> REGULARIZATION (FILE: file_name) (BETA: beta)
>> RELATIVE PERMEABILITY FUNCTION
>> SCALE
>> SCALING
>> SELEC
>> SHEAR
>> SHIFT
>> SKEWNESS
>> SKIN
>> STRAIN
>> TIME
>> TORTUOSITY
>> USER (: anno)
>> VOID FRACTION
>> YIELD
>> YOUNG

   >>> DEFAULT
   >>> LIST
   >>> MATERIAL: mat_name (mat_name_i...) (+ iplus)
   >>> MODEL
   >>> NONE
   >>> ROCK: mat_name (mat_name_i...) (+ iplus)
   >>> SET: iset
   >>> SINK: sink_name (sink_name_i ...) (+ iplus)
   >>> SOURCE: source_name (source_name_i ...) (+ iplus)
       >>>> ANNOTATION: anno
       >>>> BOUND: lower  upper
       >>>> DEVIATION: sigma
       >>>> FACTOR
       >>>> GAUSSIAN
       >>>> GUESS: guess
       >>>> INACTIVE
       >>>> INDEX: index (index_i ...)
       >>>> LOGARITHM
       >>>> LOG(F)
       >>>> NORMAL
```

```
>>>> PARAMETER: index (index_i ...)
>>>> PERTURB: (-)alpha (%)
>>>> PRIOR: prior_info
>>>> RANGE: lower  upper
>>>> STEP: max_step
>>>> TRIANGULAR
>>>> UNIFORM
>>>> VALUE
>>>> VARIANCE: sigma^2
>>>> VARIATION: sigma
>>>> WEIGHT: 1/sigma
```

```
> OBSERVATION
  >> CONCENTRATION (comp_name/COMPONENT: icomp)
                   (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> CONTENT (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> COVARIANCE (FILE: filename)
  >> CUMULATIVE (comp_name/COMPONENT: icomp)
                (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> DRAWDOWN (phase_name/PHASE: iphase)
  >> ENTHALPY (phase_name/PHASE: iphase) (CHANGE/DELTA) (WELLHEAD)
  >> FLOW (phase_name/PHASE: iphase)
          (component_name/COMPONENT: icomponent) (HEAT)
          (CHANGE/DELTA)
  >> GENERATION (comp_name/COMPONENT: icomp)
                (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> HUMIDITY (CHANGE/DELTA)
  >> MASS FRACTION (comp_name/COMPONENT: icomp)
                   (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> MOLE FRACTION (comp_name/COMPONENT: icomp)
                   (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> MOMENT (FIRST/SECOND) (X/Y/Z) (CHANGE/DELTA)
       (comp_name/COMPONENT:icomp) (phase_name/PHASE: iphase)
  >> PEST (CHANGE/DELTA)
  >> PRESSURE (CAPILLARY) (CHANGE/DELTA) (WELLHEAD/BOTTOMHOLE)
              (phase_name/PHASE: iphase)
  >> PRODUCTION (comp_name/COMPONENT: icomp)
                (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> REGULARIZATION (FILE: file_name) (BETA: beta) (CHANGE/DELTA)
  >> RESTART TIME: ntime (time_unit) (NEW)
  >> SATURATION (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> SECONDARY (phase_name/PHASE: iphase) (: ipar) (CHANGE/DELTA)
  >> STEAM QUALITY (CHANGE/DELTA)
  >> TEMPERATURE (CHANGE/DELTA) (WELLHEAD)
  >> TIME: ntime (EQUAL/LOGARITHMIC) (time_unit)
  >> TIMES from DATA/OBSERVATIONS
  >> TOTAL MASS (comp_name/COMPONENT: icomp)
                (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> USER (: anno) (CHANGE/DELTA)
  >> VOLUME (phase_name/PHASE: iphase) (CHANGE/DELTA)
  >> WATERTABLE (CHANGE/DELTA)

     >>> CONNECTION: elem1 elem2 (elem_i elem_j ...)
                     (++/+-/-+ iplus)
     >>> CONNECTION COORDINATES
         (BOX/ELLIPSOID/CYLINDER/CUBE) (ROTATE) (OUTSIDE):
          X1 Y1 Z1 (X2 Y2 Z2 (R)/(AZIMUTH DIP PLUNGE))
```

```
>>> CONNECTION PROFILE/CROSS-SECTION/MAP
>>> ELEMENT: elem (elem_i ...) (+ iplus)
>>> ELEMENT COORDINATES
   (BOX/ELLIPSOID/CYLINDER/CUBE) (ROTATE) (OUTSIDE):
     X1 Y1 Z1 (X2 Y2 Z2 (R)/(AZIMUTH DIP PLUNGE))
>>> ELEMENT PROFILE/CROSS-SECTION/MAP
>>> MODEL
>>> NONE
>>> SINK: sink_name (sink_namei ...) (+ iplus)
>>> SOURCE: source_name (source_namei ...) (+ iplus)

    >>>> ABSOLUTE
    >>>> ANNOTATION: anno
    >>>> AUTO (ADD NOISE)
    >>>> AVERAGE (VOLUME)
    >>>> BOX-COX: lambda
    >>>> COLUMN: itime idata (istd_dev)
    >>>> COMPONENT comp_name/: icomp
    >>>> DATA (NO DATA/ZERO/DATA/FILE: file_name) (time_unit)
    >>>> DETREND: timewindow (time_unit) / -npoints
    >>>> DEVIATION: sigma (ADD NOISE)
    >>>> FACTOR: factor
    >>>> FORMAT: format
    >>>> HEADER: nskip
    >>>> INDEX: index (index_i ...)
    >>>> KURTOSIS
    >>>> LOGARITHM
    >>>> MEAN (VOLUME)
    >>>> PARAMETER: index (index_i ...)
    >>>> PHASE phase_name/: iphase
    >>>> PICK: npick
    >>>> POLYNOM: idegree (time_unit)
    >>>> POTENTIAL
    >>>> PREDICTION
    >>>> REGRESSION: rho
    >>>> RELATIVE: rel_err (%) (+ const_err) (ADD NOISE)
    >>>> SET: iset
    >>>> SHIFT: shift (TIME (time_unit))
    >>>> SKEWNESS
    >>>> SKIP: nskip
    >>>> SUM
    >>>> USER
    >>>> VARIANCE: sigma^2 (ADD NOISE)
    >>>> WEIGHT: 1/sigma (ADD NOISE)
    >>>> WINDOW (INDIVIDUAL/: time_A time_B (time_unit))
```

```
> COMPUTATION

  >> CONVERGE/STOP/TOLERANCE

     >>> ADJUST
     >>> ABORT (NO)
     >>> CONSECUTIVE: max_iter1
     >>> DELTFACT: deltfact
     >>> DIRECT
     >>> FORWARD
     >>> INCOMPLETE: max_incomplete
     >>> INPUT
     >>> ITERATION: max_iter
     >>> LEVENBERG: lambda
     >>> LIST
     >>> MARQUARDT: nue
     >>> REDUCTION: max_red
     >>> SIGNAL
     >>> SIMULATION: mtough2
     >>> STEP (CURRENT/UNSCALED): max_step
     >>> UPHILL: max_uphill
     >>> WARNING

  >> ERROR
     >>> ALPHA: alpha (%)
     >>> EMPIRICAL (MATRIX: ndim (iTOUGH2)) (CORRELATION)
     >>> FISHER
     >>> FOSM (MATRIX: ndim (iTOUGH2)) (CORRELATION) (DIAGONAL)
     >>> HESSIAN
     >>> LATIN HYPERCUBE (CORRELATION/COVARIANCE) (DIAGONAL)
                         (MATRIX: ndim (iTOUGH2))
     >>> LINEARITY (: alpha (%))
     >>> LIST
     >>> MONTE CARLO (SEED: iseed) (GENERATE) (CLASS: nclass)
     >>> POSTERIORI
     >>> PRIORI
     >>> TAU: (-)niter

  >> JACOBIAN
     >>> CENTER
     >>> FORWARD (: iswitch)
     >>> HESSIAN
     >>> LIST
     >>> PERTURB: (-)perturb (%)
```

```
>> OPTION
    >>> ANDREWS: c
    >>> ANNEAL
        >>>> ITERATION: max_iter
        >>>> SCHEDULE: beta
        >>>> STEP: max_step
        >>>> TEMPERATURE: (-)temp0

    >>> CAUCHY
    >>> DESIGN
    >>> DIRECT
    >>> FORWARD
    >>> GAUSS-NEWTON
    >>> GRID SEARCH (: ninval1 (ninval2 (inval3)) /
                    FILE: filename) (UNSORTED)
    >>> L1-ESTIMATOR
    >>> LEAST-SQUARE
    >>> LEVENBERG-MARQUARDT (IDENTITY/EIGENVALUE)
                            (SUPER/TRUNCATED (: (-)truncation))
    >>> OBJECTIVE (: ninval1 (ninval2 (inval3))
                  FILE: filename) (UNSORTED)
    >>> PARALLEL: ncores (JACOBIAN / LEVENBERG: ncoreslm)
                  (SLEEP: isleep)
    >>> PEST
        >>>> DECPOINT: POINT/NOPOINT
        >>>> EXECUTABLE: file (BEFORE/AFTER)
        >>>> INSTRUCTION: num_instruction_files
        >>>> PRECISION: SINGLE/DOUBLE
        >>>> TEMPLATE: num_template_files
    >>> PVM: nhosts (JACOBIAN / LEVENBERG: nprocslm)
             (SLEEP: isleep) (FILE: node-file)
    >>> QUADRATIC-LINEAR: c
    >>> SELECT/SUPER
        >>>> CORRELATION: (-)rcorr
        >>>> IMMOBILIZATION (: ofredmin)
        >>>> ITERATION: niter
        >>>> SENSITIVITY: (-)rsens
        >>>> TRUNCATE (: (-)truncation)

    >>> SENSITIVITY
    >>> SENSITIVITY MORRIS
        >>>> PATH: npath
        >>>> PARTITION: npart
        >>>> RESAMPLE / ACCEPT
```

```
   >>> SENSITIVITY SALTELLI/SOBOL
       >>>> SAMPLES: nsamples

   >>> SEP (KURTOSIS: sepkurt) (SKEWNESS: sepskew)
   >>> SIMPLEX
   >>> STEADY-STATE (SAVE) (: (-)time_step)
   >>> SVD (TRUNCATED (: truncation))
   >>> TORNADO (: n_std_dev / BOUNDS)
   >>> WORTH (PARAMETER/PREDICTION) (DETERMINANT/TRACE) (SET)
       (METRIC: imetric) (PERCENT)

>> OUTPUT
   >>> BENCHMARK
   >>> CHARACTERISTIC
   >>> COVARIANCE
   >>> DETREND
   >>> FORMAT: format (LIST)
   >>> IDENTIFABILITY
   >>> INDEX
   >>> JACOBIAN
   >>> OBJECTIVE
   >>> PERFORMANCE
   >>> PLOTFILE: format (LIST)
   >>> PLOTTING: niter
   >>> PRINTOUT: level
   >>> RESOLUTION
   >>> SENSITIVITY
   >>> SEC/MINUTE/HOUR/DAY/WEEK/MONTH/YEAR/DATE
   >>> RESIDUAL
   >>> VERSION
   >>> WORTH (PARAMETER/PREDICTION) (DETERMINANT/TRACE) (SET)
       (METRIC: imetric) (PERCENT)
```