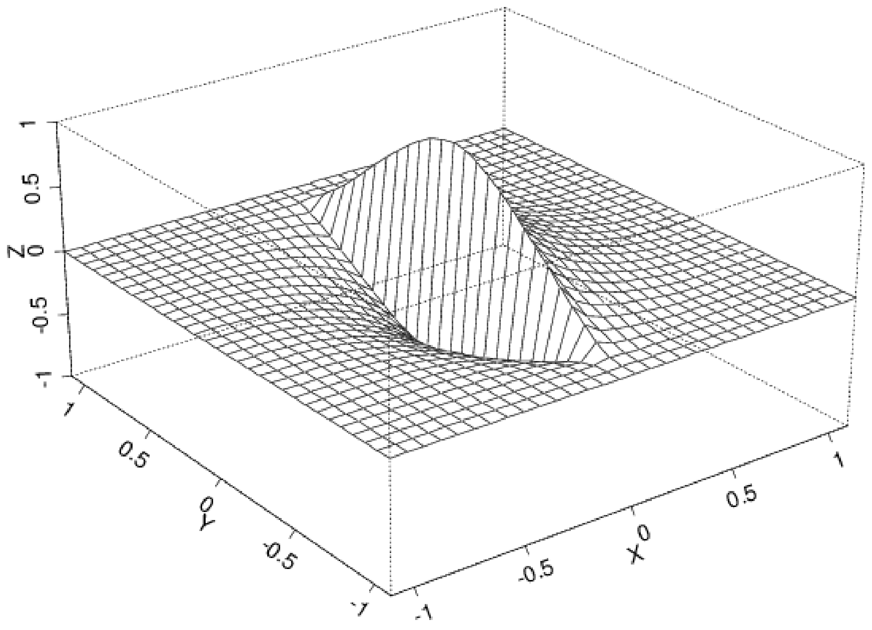
Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

Note

HAVANA user manual

Version 6.3 - Development Release



Note no SAND/27/11
Authors Per Røe
Frode Georgsen
Anne Randi Syversveen
Maria Vigsnes
Date December 22, 2011

The authors

Several people are, or have been, involved in the development of Havana at NR, including Kristin L. Munthe, Petter Mostad, Geir Aamodt, Jon Gjerde, Bjørn Fredrik Nielsen, Oddvar Lia, Knut Utne Hollund, Ariel Vazquez Almendral, Christian Skaug, Harald H. Soleng, Per Røe, Frode Georgsen, Bjørn Fjellvoll, Anne Randi Syversveen and Maria Vigsnes.

Norsk Regnesentral
Norwegian Computing Center
Postboks 114, Blindern
NO-0314 Oslo, Norway

Besøksadresse
Office address
Gaustadalléen 23
NO-0373 Oslo, Norway

Telefon · telephone
(+47) 22 85 25 00
Telefaks · telefax
(+47) 22 69 76 60

Internett · internet
www.nr.no
E-post · e-mail
nr@nr.no

Title	**HAVANA user manual**
Authors	**Per Røe , Frode Georgsen , Anne Randi Syversveen , Maria Vigsnes**
Date	December 22, 2011
Publication number	SAND/27/11

Abstract

HAVANA is a program for simulating subseismic faults in petroleum reservoirs, and for integrating the effects of these faults into the reservoir description. The HAVANA project has a long history, the original sponsors being Statoil, BP, and Norsk Hydro. Other sponsors include Conoco Norge AS, Saga Petroleum AS and Centre for Integrated Petroleum Research at University of Bergen.

Keywords	fault, simulation, sealing, stochastic, software
Target group	HAVANA users
Availability	Open
Project	HAVANA
Project number	388017
Research field	Fault modelling
Number of pages	74
Copyright © 2011	Norwegian Computing Center

Contents

1 User Reference

HAVANA is run from the UNIX command line by writing “havana” directly followed by the name of the model file. If no model file name is given, the program will look for a file named “havana.model” in the current directory and use this file, if possible.

The program is fully controlled by commands and arguments/parameters in one or more model files.

1.1 Model file syntax

Comments can be inserted into model files using ’!’ (for single line comments) or pairs of ’< and ’>’ (for multi-line comments). All lines containing a ’!’ are regarded as blank from that point on. Everything starting with a ’<’ and ending with a ’>’ is also disregarded.

All non-blank entries must be commands: They must start with a command word and end with a ’\’ character. Between the command word and the backslash is the argument list, which in general may contain any number of arguments.

The command ACTION divides the model file into sections. Before the first ACTION command comes a section with some commands pertaining to the whole program, see Section 1.2. The allowed commands in each following section depends on the specific ACTION command heading it.

Within each section, the commands may appear in any order, but any command (except READ commands) may appear at most once. READ commands make it possible to read commands from other files, so that one may split the model file into several smaller files (see Section 1.2.4).

The different sections correspond to different actions within HAVANA. Each module is independent of the others in that it reads all of its input from files, and writes all results to files before the next module is started. This is slightly inefficient, but has the advantage that HAVANA may be restarted at any module that causes it to terminate, instead of at the very beginning.

The whole model file is checked before any part of the actual program is run. If any errors are found, they are listed, and the program is terminated. Note that there may still be errors in the input files read by the program: It is not checked that for example ECLIPSE input files or RMS input files are correct before the program execution reaches the place where they are read in.

All work flows starts with ImportRMSFaultData. All communication between the actions goes through the internal Havana format, described in A.12. The communication is handled with the commands input/output original/modified havana faults, see commands 1.3.8, 1.4.7 and 1.4.8.

1.2 Commands common to the whole program

Many HAVANA commands may only appear in specific sections, i.e., below specific ACTION commands. However, some commands may appear before any action commands, or in any section, or indeed anywhere. These are listed below.

1.2.1 Command: INPUT_AND_OUTPUT_DIRECTORY

Optional: May only appear before any of the ACTION commands.

Description: All files read and written by the program will be assumed located relative to this directory.

Arguments: One. The name of a directory, either relative to the one where the program is run, or absolute.

Examples:

```
INPUT_AND_OUTPUT_DIRECTORY ../mydir \
```

```
INPUT_AND_OUTPUT_DIRECTORY /user/geir/havana/example/ \
```

1.2.2 Command: INPUT_DIRECTORY

Optional: May only appear in the ACTION sections.

Description: This command overrides any INPUT_AND_OUTPUT_COMMAND with relation to INPUT files, and redirects the search for input files in the current ACTION section.

Arguments: One. The name of a directory, either relative to the one where the program is run, or absolute.

Example:

```
INPUT_DIRECTORY newfield/data \
```

1.2.3 Command: OUTPUT_DIRECTORY

Optional: May only appear in the ACTION sections.

Description: This command overrides any INPUT_AND_OUTPUT_COMMAND with relation to OUTPUT files, and redirects the writing of output files in the current ACTION section.

Arguments: One. The name of a directory, either relative to the one where the program is run, or absolute.

Example:

```
OUTPUT_DIRECTORY results \
```

1.2.4 Command: READ

Optional: May appear anywhere, any number of times.

Description: This command includes any set of valid HAVANA commands written in a separate file. The effect is as if the commands had appeared directly in the main model file. Any commands may appear in such files (including new READ commands) *except* ACTION commands. This guarantees that one will always know what modules are run just by reading the top model file.

Arguments: The name of one or more files with HAVANA commands.

Examples:

```
READ simulate.model \
```

```
READ sim1.model sim2.model \
```

1.2.5 Command: SEED

Optional: If neither a seed number or a seed file are available, a random seed number will be used. If both are available, the seed number given in the seed file is used.

Description: Sets the seed for the random number generator.

Arguments: One. An integer between 0 and 4294967295.

Example: `SEED 74839254 \`

1.2.6 Command: INPUT_SEED_FILE

Optional: If neither a seed number or a seed file are available, a random seed number will be used. If both are available, the seed number given in the seed file is used.

Description: Reads and sets the seed for the random number generator from a file. If the file does not exists, it will be created. The value of the seed after the simulation is written out to the file.

Arguments: One. The name of an ASCII file containing one single integer between 0 and 4294967295.

Example: `INPUT_SEED_FILE seed.dat \`

1.2.7 Command: LEVEL_OF_INFORMATION

Optional: May appear anywhere. If it appears before any ACTION command, it affects the whole program, while if it appears below an ACTION command, it affects only that section.

Description: The command regulates the amount of information output to screen and to the log file. There are six levels of information: 0, 1, 2, 3, 4 and 5. If level 0 is used, only warnings and errors are written. Level 4 and 5 give debug information. When LEVEL_OF_INFORMATION does not appear, information level 0 is used. Note that even if the information level is zero, information at level 1 will be output to the log file.

Arguments: One. An integer 0, 1, 2, 3, 4 or 5. If LEVEL_OF_INFORMATION appears without any argument, it corresponds to information level 1.

Example: `LEVEL_OF_INFORMATION 2 \`

1.2.8 Command: OUTPUT_LOG

Optional: May only appear before any ACTION commands.

Description: Causes information from the program to be saved in a log file. The information is the same as that appearing on the screen, but if the information level is zero, information at level 1 will still be output to the log file.

Arguments: One. The name of the log-file.

Example: `OUTPUT_LOG logfile.dat \`

1.3 ImportRMSFaultData: Input of fault model/structural model from RMS

ACTION ImportRMSFaultData \
This action has several commands used to import fault data from RMS

There are two ways to export fault data from the integrated structural model in RMS. A manual export of the different types of data which is available in RMS 2009 and newer, and a single click export of a complete structural model which currently only is available in special development versions of RMS 2011.

With old RMS, the command INPUT_FAULTS specifies where the fault data are located. Additionally the fault lines have to be given with the FAULT_LINE_POLYGON command. For details about this export format, see A.2.

When using the new export format, the directory containing the structural model is given by the INPUT_STRUCTURAL_MODEL command. The command FAULT_LINE_POLYGON is not in use, except for giving additional input data sets in the UpdatePoints action. For details about this export format, see A.1.

The directory containing the fault model file, and the fault surfaces is given by the INPUT_FAULTS command, the fault displacement is deduced from the fault lines given by the FAULT_LINE_POLYGON command, additional parameters needed to generate the displacement field are given by the SFM_PARAMETERS, VARIO_TYPE, FAULT_DISPLACEMENT_LENGTH and FAULT_LENGTH_HEIGHT keywords. The type of fault given in SFM_PARAMETERS is ignored in the new export format since the same information about whether a fault is normal or reverse is exported directly by RMS in the fault_model file.

1.3.1 Command: INPUT_FAULTS

Necessary: Only used for import of faults from RMS on old format.

Description: Specifies the name of the directory containing the fault files. The directory must contain a faultmodel.txt file together with the fault surfaces as point sets.

To obtain the faultmodel.txt file the **RMS_FAULT_MODEL_FILENAME** environment variable should be set to the desired location of this file. The file is then generated whenever a **Fault modelling** job is run within RMS.

The fault surfaces as point sets are generated by extracting the fault surfaces from the structural model within RMS and export the resulting point sets using the **Roxar text** format.

Arguments: One. The directory.

Example:

```
INPUT_FAULTS faults/sfm_faults \
```

1.3.2 Command: FAULT_LINE_POLYGON

Necessary: If the faults are given on the old format.

Description: List of files with fault line polygons exported from RMS. One file corresponds to one horizon, and contains fault lines for more than one fault.

The fault lines are generated in RMS by extracting fault lines from the horizon model within the structural model. The resulting fault lines are placed in the Horizons list and exported using the **Roxar text** format. Files of type RMSInternalPoints (control points) can also be read. Then information about hw/fw and fault name are also given in the file, and this information is also read and used.

Arguments: One or more. Each argument specifies a file containing a set of fault line polygons.

Example:

```
FAULT_LINE_POLYGON
TopC_faultlines.points ;
BaseA_faultlines.points \
```

1.3.3 Command: INPUT_STRUCTURAL_MODEL

Necessary: Only used for import of data on new RMS export format.

Description: Specifies the name of the directory containing the fault files when using new RMS. The directory must contain a fault_model file together with the fault surfaces as point sets.

To obtain the fault_model file use the command ExtractFaultLinesToFile under StructuralModels/Horizons in RMS.

Arguments: One. The directory.

Example:

```
INPUT_STRUCTURAL_MODEL faults/sfm_faults \
```

1.3.4 Command: SFM_PARAMETERS

Necessary

Description: Parameters describing the fault and corresponding influence area.

Arguments A list of faults with corresponding parameters. For each fault the fault name, fault throw distribution (=1.0 all throw is distributed on hanging wall side, =0.0 all throw is distributed on foot wall), reverse drag distance (measured laterally) and whether the fault is normal (=1) or reverse (=-1) is given. For faults exported on the new format the information whether a fault is normal or reverse is exported from RMS in the fault_model file.

If the name 'default' is given the following parameters applies to all faults with no explicitly given parameters.

Example:

```
SFM_PARAMETERS
F3 1.0 500 1 ;
F1 0.7 2000 -1 ;
default 0.5 1000 1 \
```

1.3.5 Command: VARIO_TYPE

Optional: Otherwise, default values are used.

Description: Specifies a variogram model used when modeling the displacement field on the fault surface.

Default is a spherical variogram without any anistrophy, with range 1000 in strike direction and range 2000 in dip direction.

Arguments: Four or five:

1. Variogram type. Possible variogram types are:
   - GAUSSIAN - Gaussian variogram.
   - SPHERICAL - Spherical variogram.
   - EXPONENTIAL - Exponential variogram.
   - GENERAL_EXPONENTIAL - General exponential variogram. An additional parameter giving the power must be given.
2. Range in strike direction. Higher range gives smoother fault surfaces.
3. Range in dip direction. Higher range gives smoother fault surfaces.
4. Anisotrophy angle in degrees. Use this if you want to rotate the direction of the ranges.
5. The power. Only for general exponential variograms.

Examples:

```
VARIO_TYPE SPHERICAL 1000 500 0.0 \
```

1.3.6 Command: FAULT_DISPLACEMENT_LENGTH

Optional: Otherwise, default values are used.

Description: The relationship between maximum displacement and length of fault.

Arguments: Two constants, a and b, where $length = (maximumDisplacement/b)^{(1/a)}$

Default values, if command is not defined, are 1.0 and 0.01.

Examples: `FAULT_DISPLACEMENT_LENGTH 1.0 0.01 \`

1.3.7 Command: FAULT_LENGTH_HEIGHT

Optional: Otherwise, default values are used.

Description: The relationship between length and height of fault.

Arguments: One constant, c, where $height = (length/c)$.

Default value, if command is not defined, is 2.0.

Examples: `FAULT_LENGTH_HEIGHT 2.0 \`

1.3.8 Command: OUTPUT_ORIGINAL_HAVANA_FAULTS

Necessary

Description: Specifies the name of the directory where the original faults are written to. These faults are written on the internal Havana format (See A.12)

Arguments: One. The directory.

Example: `OUTPUT_ORIGINAL_HAVANA_FAULTS originalHavanaFaults  \`

1.4 FaultUncertaintyEnvelope: Define an uncertainty envelope around fault

ACTION FaultUncertaintyEnvelope \
This action is used to read pointsets from file and interpolate these points to get an uncertainty envelope around the fault consisting of one surface on the hanging wall side and one surface on the footwall side of the fault surface. For a fault uncertainty envelope to be generated, either command 1.4.2 or 1.4.3 or both must be given.

Here are the available commands:

1.4.1 Command: VARIOGRAM_GEOMETRY

Optional: Otherwise, default values are used.

Description: Variogram used when simulating or predicting the fault surface.

Default is a gaussian variogram without any anisotrophy, with range 1000 in strike direction and range 2000 in dip direction.

Arguments: Four or five.

1. Variogram type. Possible variogram types are:
   - GAUSSIAN - Gaussian variogram.
   - SPHERICAL - Spherical variogram.
   - EXPONENTIAL - Exponential variogram.
   - GENERAL_EXPONENTIAL - General exponential variogram. An additional parameter giving the power must be given.
2. Range in strike direction. Higher range gives smoother fault surfaces.
3. Range in dip direction. Higher range gives smoother fault surfaces.
4. Anisotrophy angle in degrees. Use this if you want to rotate the direction of the ranges.
5. The power. Only for general exponential variograms.

Example:
```
VARIOGRAM_GEOMETRY   GAUSSIAN 1000 2000 0.0 \
```

1.4.2 Command: INPUT_ENVELOPE_POINTS

Optional

Description: Specifies on each line a fault name and a file with points that is used to define the uncertainty envelope. There can be an arbitrary number of lines and the fault names can be repeated meaning that point sets from several files for the same fault are pooled into one big set.

Arguments: Two on each line. The fault name and the file name containing a point set.

Example:
```
INPUT_ENVELOPE_POINTS
CMQ_main RMS_CMQ_LateralUncertainty.dat \
```

1.4.3 Command: CONSTANT_ENVELOPE

Optional

Description: Specifies on each line a fault name and a constant distance away from the fault along the normal vector that defines the uncertainty envelope on each side of the fault surface. If the name 'default' is given the following value applies to all faults with no explicitly given value.

Arguments: Two on each line. The fault name (or 'default') and the value for the uncertainty distance.

Example:

```
CONSTANT_ENVELOPE
F2 350 ;
default 250 \
```

1.4.4 Command: FAULT_PICKS_FILE

Optional

Description: File with well observations of faults. Used for conditioning on fault surface and uncertainty envelope.

Arguments: One. Name of file containing fault picks on RMS format.

Example:
```
FAULT_PICK_FILE   wellPicks.txt \
```

Example of fault picks file:

```
F1  Well-9   169800.000   570800.000    1727.5908
F10 W8       169047.814   570891.481    1789.8117
F9  W1       169958.321   569800.000    1832.5233
```

1.4.5 Command: FAULT_PICK_UNCERTAINTY

Optional: Only relevant if command 1.4.4 is given.

Description: Common uncertainty for all given fault picks. The value gives the maximum distance away from the fault pick along the faults' normal vector that the fault is allowed to vary. The uncertainty envelopes are conditioned to the point that is located this distance away from the fault pick on each side of the fault. If the given uncertainty envelope is closer to the fault pick that this distance the closest point is used in the conditioning.

Arguments: One. Uncertainty in fault picks.

Example:
```
FAULT_PICK_UNCERTAINTY   20 \
```

1.4.6 Command: OUTPUT_FAULTS_ENVELOPE

Optional

Description: Specifies the name of the directory where the faults with their uncertainty envelopes are written to. Only faults that have an envelope on at least one side are written. The fault surface is also written. The surfaces are written on Roxar Text format.

Arguments: One. The directory.

Example:
```
OUTPUT_FAULTS_ENVELOPE Uncertainty   \
```

1.4.7 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

Description Specifies the name of the directory where the original faults on the internal Havana format (See A.12) are read from.

Arguments: One. The directory.

Example:
```
INPUT_ORIGINAL_HAVANA_FAULTS originalHavanaFaults  \
```

1.4.8 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Necessary

Description: Specifies the name of the directory where the modified faults are written to. These faults are written on the internal Havana format (See A.12).

Arguments: One. The directory.

Example:
```
OUTPUT_MODIFIED_HAVANA_FAULTS modifiedHavanaFaults  \
```

1.5 ModifyFaultSurface: Rotate or move fault surface

ACTION ModifyFaultSurface \
This action is used to modify the fault surface of a set of faults on the SFM format. The commands 1.5.6, 1.5.7 and 1.5.8 can be given independently of each other for various (or the same) faults. The modifications are performed in the listed order.

Here are the available commands:

1.5.1 Command: VARIOGRAM_GEOMETRY

Optional: Otherwise, default values are used.

See 1.4.1.

1.5.2 Command: FAULT_PICKS_FILE

Optional

See 1.4.4.

1.5.3 Command: OUTPUT_FAULTS

Necessary

Description: Specifies the name of the directory where faults are written to.

Arguments: One. The directory.

Example:
```
OUTPUT_FAULTS transfsurfaceFaults  \
```

1.5.4 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7.

1.5.5 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Optional

See 1.4.8.

1.5.6 Command: LOCAL_ROTATION

Optional

Description: Specifies rotations performed on the faults by changing azimuth and/or dip angle.

Arguments: At least three. Name of subcommand, name of fault, change of angle given in degrees. Possible subcommands are CHANGE_STRIKE and CHANGE_DIP

Note: The fault lines and horizons (if specified) on output are not changed according to the rotations.

Example:

```
LOCAL_ROTATION
CHANGE_STRIKE F2 45 ;
CHANGE_DIP F1 90  \
```

1.5.7 Command: TRANSFORM

Optional

Description: Specifies transformations performed on the faults.

Arguments: At least three. Name of subcommand, name of fault, distance for translation. Possible subcommands are TRANSLATE_X (translation parallel to the global x-axis), TRANSLATE_Y (translation parallel to the global y-axis), TRANSLATE_NORM (translation parallel to the faults normal vector projected to the global xy-plane).

Note: The fault lines and horizons (if specified) on output are not changed according to the transformations.

Example:

```
TRANSFORM
TRANSLATE_NORM F1 1000 ;
TRANSLATE_X F2 300 \
```

1.5.8 Command: TRANSLATE_NORM_RELATIVE

Optional

Description: Translates the fault parallel to the normal vector projected to the global xy-plane with a factor relative to the distance to the fault uncertainty envelope on the relevant side of the fault.

Arguments: Two. Fault name and factor. The factor must be between -1 and 1, with negative number meaning moving in hanging wall direction and positive number meaning moving in foot wall direction.

Note: Fault uncertainty envelopes must be defined by the action FaultUncertaintyEnvelope (see 1.4). In case these envelopes are not properly defined, the translation is ignored.

Example:

```
TRANSLATE_NORM_RELATIVE
F1 0.5 ;
F3 -0.3  \
```

1.6 SimulateFaultSurface: Simulate fault surface

ACTION SimulateFaultSurface \
This action is used to generate stochastic realisations of a set of faults on the SFM format. Note that the Action FaultUncertaintyEnvelope must be performed prior to this action in order to generate uncertainty volumes around the faults that are going to be simulated.

Here are the available commands:

1.6.1 Command: OUTPUT_FAULTS

Necessary

See 1.5.3.

If more than one realization is simulated, underscore and the realization number is added to the directory name.

1.6.2 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7

1.6.3 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Optional

See 1.4.8

If more than one realization is simulated, underscore and the realization number is added to the directory name.

1.6.4 Command: VARIOGRAM_GEOMETRY

Optional: Otherwise, default values are used.

See 1.4.1.

1.6.5 Command: DISTRIBUTION

Optional: Otherwise, default distribution is used.

Description: Distribution for points on fault surface within fault volume. Uniform and triangular distributions ensure that the simulated fault surface is within the given fault volume. Fault volume is given in command SIMULATION.

Uniform distribution is used as default.

Arguments: : One. Name of distribution. There are three possible distributions:

- UNIFORM - Uniform distribution within given fault volume.
- TRIANGULAR - Triangular distribution within given fault volume with mode equal to base case.
- NORMAL - Normal distributed with mean equal to base case, and standard deviation equal to half the distance to the border of the fault volume.

Example:
```
DISTRIBUTION   UNIFORM \
```

1.6.6 Command: NUMBER_OF_REALIZATIONS

Optional: Otherwise, default is used.

Description: Number of realizations.

As default one realization is made.

Arguments: : One - the number of realizations requested.

Example:
```
NUMBER_OF_REALIZATIONS   5 \
```

1.6.7 Command: SIMULATE_FAULTS

Optional: Otherwise, default is used.

Description: List of faults that should be simulated.

As default all faults in input fault set will be simulated.

Arguments: List of fault names.

Example:
```
SIMULATE_FAULTS
F1 F3 \
```

1.7 ModifyDisplacement: Scale or add throw to displacement field

ACTION ModifyDisplacement \
This action is used to modify the displacement field of a set of faults on the SFM format. The commands 1.7.7 and 1.7.8 can be given independently of each other for various (or the same) faults. The modifications are performed in the listed order.

Here are the available commands:

1.7.1 Command: VARIO_TYPE

Optional: Otherwise, default values are given.

See 1.3.5.

1.7.2 Command: FAULT_DISPLACEMENT_LENGTH

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.6.

1.7.3 Command: FAULT_LENGTH_HEIGHT

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.7.

1.7.4 Command: OUTPUT_FAULTS

Necessary

See 1.5.3

1.7.5 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7

1.7.6 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Optional

See 1.4.8

1.7.7 Command: SCALE_DISPLACEMENT

Optional

Description: Specifies the factor for the change of displacement .

Arguments: At least two. Name of fault, multiplier for displacement change. The displacement at every point on the fault surface is multiplied by this factor.

Result: A modified displacement grid for each fault is written to a file in the directory specified in the command OUTPUT_MODIFIED_HAVANA_FAULTS. This modified fault is used in the Action UpdatePoints.

Example:

```
SCALE_DISPLACEMENT
F3 1.2 ;
```

```
        F1 0.7 \
```

1.7.8 Command: ADD_THROW

Optional

Description: Specifies a number for the change of throw by adding this number .

Arguments: At least two. Name of fault, constant for throw addition. The throw at every point on the fault surface is increased by this factor. If the number is positive the foot wall side is moved up and hanging wall side is moved down. If the number is negative, foot wall side is moved down and hanging wall side moved up.

Result: A modified displacement grid for each fault is written to a file in the directory specified in the command OUTPUT_MODIFIED_HAVANA_FAULTS. This modified fault is used in the Action UpdatePoints.

Example:

```
        ADD_THROW
        F2 10 ;
        F4 -5.4 \
```

1.8 SimulateDisplacement: Fault displacement field modelling

ACTION SimulateDisplacement \
In this action the fault displacement fields and the corresponding fault tip lines can be predicted or simulated from a set of input displacement observations. The necessary and optional keywords for the SimulateDisplacement action are described here.

1.8.1 Command: VARIO_TYPE

Optional: Otherwise, default values are given.

See 1.3.5.

1.8.2 Command: FAULT_DISPLACEMENT_LENGTH

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.6.

1.8.3 Command: FAULT_LENGTH_HEIGHT

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.7.

1.8.4 Command: OUTPUT_FAULTS

Necessary

See 1.5.3.

1.8.5 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7.

1.8.6 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Optional

See 1.4.8.

1.8.7 Command: SEISMIC_RESOLUTION

Necessary

Description: The seismic resolution, which defines standard deviation of uncertainty in observations.

Arguments: One. One constant value.

Example:
```
SEISMIC_RESOLUTION 10.0 \
```

1.8.8 Command: SIMULATION

Optional: Otherwise, default is used.

Description: Indicator telling whether we should simulate or predict or leave the fault unchanged.

Default is to simulate all faults, i.e. 1 for all faults.

Arguments: One for each fault. 1 for simulation, 0 for prediction, -1 for unchanged. Default value is 1. If the name 'default' is given the following parameter applies to all faults with no explicitly given parameter.

Example:

```
SIMULATION
F1 0 ;
F3 -1 ;
default 1 \
```

1.9 RemoveFaults: Remove faults from fault set

ACTION RemoveFaults \
Here are necessary and optional keywords for the action RemoveFaults. The action takes the faults specified in the keyword REMOVE_FAULT, and removes it from the fault set. The modified fault set without the removed faults is written to file.

1.9.1 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7.

1.9.2 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Necessary

See 1.4.8

1.9.3 Command: OUTPUT_FAULTS

Optional

See 1.5.3.

1.9.4 Command: REMOVE_FAULT

Neccesary

Description: Gives the name of the faults that is going to be removed from the original fault set.

Arguments: One (the name) for each fault.

Example:

```
REMOVE_FAULT
F1 ;
S2 \
```

1.10 AddFaults: Add Faults to surface

ACTION AddFaults \

This action is used to include new faults to the original fault set in a modified fault set. New faults can be included as elliptic faults or as fault surface extracted from RMS without displacement field. Elliptic faults that are added have specified displacement field, but the fault surfaces extracted from RMS have not. The displacement field for these faults can be set by the SetDisplacement action. New truncation rules related to the new faults are read from file and added to the fault set.

1.10.1 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7.

1.10.2 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Necessary

See 1.4.8.

1.10.3 Command: OUTPUT_FAULTS

Optional

See 1.5.3.

1.10.4 Command: ADD_ELLIPTIC_FAULTS

Optional

Description: Specifies the file name for the elliptic faults. Imports faults on simplified elliptic format. The format is described in Appendix A.3.

Arguments: One. The filename.

Example:

```
ADD_ELLIPTIC_FAULTS elliptic_faults.ell \
```

1.10.5 Command: ADD_FAULT_SURFACE

Optional

Description: Specifies the name of new fault surfaces and the name of the input files from RMS. The fault surfaces have no displacement field. The file format is described in Appendix A.1.2.

Arguments: At least two. The name of the fault surface and the filename.

Example:

```
ADD_FAULT_SURFACE
F11 F11_grid ;
F12 F12_grid  \
```

1.10.6 Command: TRUNCATION_RULES

Optional: Only necessary if the new fault introduces new truncation rules.

Description: Specifies the name of the file with new truncation rules to be added. The file format is plain ascii where each truncation rule is given by one line of three parameters, i.e.

the truncated fault, the truncating fault and the side of the truncating fault where the fault is truncated.

Arguments: One. The filename.

Example:

```
TRUNCATION_RULES truncation_rules.txt \
```

1.11 SetDisplacement: Set elliptic displacement to fault surface

ACTION SetDisplacement \
This action is used to add elliptic displacement field to existing faults or fault surfaces. The action is relevant when new fault surfaces without displacement field have been added to the fault set in the command AddFaults, however can also replace existing displacement field in any of the faults given in the fault set. The displacement field can be specified by giving the maximum displacement, which requires that fault displacement parameters are given (either specified for each fault or by giving default values). The horizon where the maximum displacement should be located can be given (optional), but requires that the faultlines are given as input file. If fault lines are given but no horizon specified, the maximum displacement is located at the centre point of all fault line points for the given fault. Otherwise, the maximum displacement is set at the centre point of the fault surface.

Alternatively the displacement field can be set by specifying the elliptic fault parameters in an input file. This requires that the centre point of the ellipse is located at or close to the fault surface of the corresponding fault.

Here are the available commands:

1.11.1 Command: INPUT_MODIFIED_HAVANA_FAULTS

Necessary

See 1.4.7.

1.11.2 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

Necessary

See 1.4.8.

Optional

1.11.3 Command: OUTPUT_FAULTS

See 1.5.3.

1.11.4 Command: SET_MAX_DISPLACEMENT

Optional

Description: Specifies maximum displacement of the displacement field to be added to the given fault. Requires command 1.11.5. If horizon name is given, the maximum displacement is set at this horizon, but requires command.1.11.8. If fault lines are given but no horizon specified, the maximum displacement is located at the centre point of al fault line points for the given fault. Otherwise it is set at the centre point of the fault surface.

Arguments: At least two. Name of fault, size of maximum displacement. Horizon name for location of maximum displacement optional. Displacement field given by maximum displacement can be given to several faults.

Example:

```
SET_MAX_DISPLACEMENT
F1  200 TopC  ;
F7  500 BaseA  \
```

1.11.5 Command: FAULT_DISPLACEMENT_PARAMETERS

Optional: Necessary when command 1.11.4 is given.

Description: Parameters describing the fault and corresponding influence area.

Arguments: A list of faults with corresponding parameters. For each fault the fault name, fault throw distribution (=1.0 all throw is distributed on hanging wall side, =0.0 all throw is distributed on foot wall), range (measured laterally) and whether the fault is normal (=1) or reverse (=-1) is given.

If the name 'default' is given the following parameters applies to all faults with no explicitly given parameters.

Example:

```
FAULT_DISPLACEMENT_PARAMETERS
F1  1.0 500  1  ;
F7  0.7 2000 -1 ;
default 0.5 1000 1   \
```

1.11.6 Command: FAULT_DISPLACEMENT_LENGTH

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.6.

1.11.7 Command: FAULT_LENGTH_HEIGHT

Optional: Otherwise, values given in a previous action applied on the same fault set are used.

See 1.3.7.

1.11.8 Command: INPUT_FAULT_LINES

Optional

Description: Specifies the file name for the fault line file. The file format is described in A.1.3.

Arguments: One. The filename.

Example:

```
INPUT_FAULT_LINES fault_lines \
```

1.11.9 Command: SET_DISPLACEMENT_FROM_FILE

Optional

Description: Specifies the filename for the elliptic displacement. Imports elliptic displacement on simplified elliptic format, and add this to the fault specified for each line. See the format described in Appendix A.3.

Arguments: One. The filename.

Example:

```
SET_DISPLACEMENT_FROM_FILE displacement_field.ell \
```

1.11.10 Command: DEBUG_FAULT_INFO

Optional

Description: Specifies the name of the directory where the debug fault information is written to. The global data and point sets for all faults are written.

Arguments: One. The directory.

Example:

```
DEBUG_FAULT_INFO faultInfo \
```

1.12 UpdatePoints: Update point sets after modifying or simulating faults

ACTION UpdatePoints \
This action is used to update points after modification of fault surface. All horizons and fault lines are updated by specifying the structural model. Only selected pointset (horizons and/or fault lines) can be updated by specifying these under input pointset. Both the structural model and pointset can be given in same model file.

If fault block definition is given (either as default in the structural model or in the fault block definition command) all horizons, faultlines and pointset given on RMS Internal Point Format will be tagged with updated smart-fault-tag.

Here are the available commands:

1.12.1 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7

1.12.2 Command: INPUT_MODIFIED_HAVANA_FAULTS

Necessary

Description Specifies the name of the directory where the modified faults on the internal Havana format are read from.

Arguments: One. The directory.

Example:
```
INPUT_MODIFIED_HAVANA_FAULTS modifiedHavanaFaults  \
```

1.12.3 Command: INPUT_STRUCTURAL_MODEL

Optional

See 1.3.3. By using this command, all fault lines and horizons are updated.

1.12.4 Command: INPUT_POINTSET

Optional

Description: Pointsets which are horizons or fault lines that should be updated according to the changes in fault displacement.

Arguments: At least one. Name of files with pointset. Legal formats are **Roxar text**, RMSInternalPoints (control points) (see section A.11) and Storm format.

Example:

```
INPUT_POINTSET horizons/topC.xyz ;
               horizons/baseA.xyz ;
               faultlines/TopCFaultLines.txt ;
               faultlines/BaseAFaultLines.txt \
```

1.12.5 Command: NODEFILE

Optional

Description A file with a point set is read and points are moved. The moved points are written to a file with name "inputfile"moved, where "inputfile" is the name of the input file.

Arguments: The name of the input file.

Example:

```
    NODEFILE nodes.dat \
```

1.12.6 Command: PRINT_ONLY_FAULTLINES_FROM_CHANGED_FAULTS

Optional

Description When this command is given, only fault lines from changed faults of the structural model are written to file.

Arguments None.

1.12.7 Command: FAULT_BLOCK_DEFINITION

Optional: Necessary if points should be tagged with smart-fault-tag when structural model is not given.

Description File which contains definition of fault blocks. To be used if horizons contain information about fault blocks.

Arguments Name of input file. The file format is plain ascii, and one line for each fault block. It starts with fault block number, then fault name and side (fw, hw) for all faults with known side. Faults with undefined side need not be mentioned. Example of fault block file is given in A.9.

Example:

```
    FAULT_BLOCK_DEFINITION faultblocks.txt \
```

1.12.8 Command: FILTER_DISTANCE

Optional

Description This argument gives a custom filtering distance for horizon and fault line points. Sometimes RMS interpret points to be on the different side as they are interpreted in Havana. This filtering removes the points that are so near a fault surface that this can be a problem.

Arguments One. Distance used for filtering horizon and fault line points. Points moved so they are within the given distance of a fault surface will be removed. Default value is 1.0.

1.12.9 Command: NOT_FILTER_POINTS_CROSSING_FAULTS

Optional

Description When this command is given, we do not filter points crossing faults when moving horizons.

Arguments None.

1.12.10 Command: FILTERING_FROM_WELLS

Optional

Description Fault line points close to well points are filtered if they are on the same side and within a certain distance from a well point. Optionally, also points on opposite side can be filtered.

Arguments Name of input file, minimum distance for filtering, and optional an indicator (0 or 1) telling if points on opposite side of the well point should be filtered. The file is a plain ascii text file, which can be exported from RMS. Each line contains first two strings which are not used, and then x, y, and z coordinates of a point.

Example:

```
FILTERING_FROM_WELLS welldata.dat 1000.0 1 \
```

1.12.11 Command: NOT_FILTER_INACTIVE_HORIZON_POINTS

Optional Default value is true if not given

Description When this command is given the horizon points that are tagged as inactive on input are updated and written to file.

Arguments None.

1.12.12 Command: NOT_FILTER_ERODED_FAULTLINE_POINTS

Optional Default value is true if not given

Description When this command is given the fault line points that are tagged as eroded on input are updated and written to file.

Arguments None.

1.12.13 Command: OUTPUT_HORIZONS

Optional: Necessary when command 1.12.3 is given.

Description: Directory to store output horizons.

Arguments: One. Name of directory.

Example:

```
OUTPUT_HORIZONS horizons \
```

1.12.14 Command: OUTPUT_FAULTLINES

Optional: Necessary when command 1.12.3 is given.

Description: Directory to store output fault lines.

Arguments: One. Name of directory.

Example:

```
OUTPUT_FAULTLINES faultLines \
```

1.12.15 Command: OUTPUT_POINTSET

Optional: Necessary when command 1.12.4 is given.

Description: Directory to store output pointset.

Arguments: One. Name of directory.

Example:

```
OUTPUT_POINTSET pointset \
```

1.13 UpdateGrid: Update grid with new faults

ACTION UpdateGrid \
This action is used update the eclipse grid to include new faults given in the modified fault set (compared to the original fault set). The fault surface is defined by grid pillar lines. The blocks affected by the new faults are divided, and displacement is added to the grid according to the displacement field of the new faults. The truncation rules are taken into account.

Here are the available commands:

1.13.1 Command: INPUT_GRID

Necessary

Description: Specifies the file of the input eclipse grid.

Arguments: One. The filename.

Example:

```
INPUT_GRID  emerald_structmodgrid1.grdecl \
```

1.13.2 Command: OUTPUT_GRID

Necessary

Description: Specifies the file name of the eclipse grid with the new faults.

Arguments: One. The filename.

Example:

```
OUTPUT_GRID output.grdecl \
```

1.13.3 Command: INPUT_ORIGINAL_HAVANA_FAULTS

Necessary

See 1.4.7.

1.13.4 Command: INPUT_MODIFIED_HAVANA_FAULTS

Necessary

See 1.4.7.

1.13.5 Command: FAULT_BLOCK_DEFINITION

Necessary

See 1.12.7.

1.14 TestFaults: Test reading and writing of faults

ACTION TestFaults \
The TestFaults action is useful in QC of the structural model and debugging of Havana. This action writes a lot of potentially useful information to the `Debug` subdirectory of the output directory.

The following information is given in the following directories:

fault_info contains a point set file for each fault on RMS internal points format. The point set have the following attributes:

- **Dip** Local dip in degrees.
- **Strike** Local strike in degrees.
- **Displacement** Local displacement attributed to the current fault.
- **TotalDisplacement** Local total displacement, both from this fault and all truncating faults. Corresponds to the observed displacement.
- **FwMoved** How much a point on the foot wall side of the fault has been moved. Used for inverse movement.
- **HwMoved** How much a point on the hanging wall side of the fault has been moved. Used for inverse movement.

displacement_field_generation contains a file for each input fault on RMS internal points format containing information used in the generation of the displacement field. This information includes:

- **DataPoints** Displacement data obtained from fault lines. A value of 0 indicates that there are no displacement data for the given point. The data points are estimated based on the restored fault lines.
- **Trend** The elliptic trend estimated from the data points.
- **Displacement** The values in the final displacement field for the fault.
- **RelDisplacement** Relative displacement, displacement scaled to a value between 0 and 10.

side_of_fault contains a grid for each fault on STORM binary format with codes telling which side of the fault the grid cells are. The values are

- **0** if the cell is outside the volume affected by the fault.
- **1** if the cell is on the hanging wall side of the fault.
- **2** if the cell is on the foot wall side of the fault.

distance_to_fault contains a grid file for each input fault. The grid is populated with distances to the fault surface for the given fault. The distance is positive on the hanging wall side of the fault, and negative on the foot wall side. For points outside the fault surface, the distance to the reference plane is given.

fault_points_sided contains the fault line points sorted per horizon, fault and side of fault. Contains two subdirectories, one with the fault lines on Roxar text format, and one with the fault lines on RMS internal points format, tagged with a segment number visualizing how Havana splits the set of fault line points according to truncations.

restored_fault_lines contains the restored fault line points in the same formats as described for fault_points_sided. In addition the fault lines for each horizon are given in Roxar text format and internal points format.

fault_surface_extrapolation contains a set of surface files for each fault visualizing the fault surface on various steps in the fault surface extrapolation.

The supported commands for the TestFaults action are given below.

1.14.1 Command: INPUT_FAULTS

Necessary: Only used for import of faults from RMS on old format.

See 1.3.1.

1.14.2 Command: FAULT_LINE_POLYGON

Necessary: Only used for import of faults from RMS on old format.

See 1.3.2.

1.14.3 Command: INPUT_STRUCTURAL_MODEL

Necessary: Only used for import of data on new RMS export format.

See 1.3.3.

1.14.4 Command: OUTPUT_FAULTS

Necessary

See 1.5.3.

1.14.5 Command: SFM_PARAMETERS

Necessary

See 1.3.4.

1.14.6 Command: VARIO_TYPE

Optional: Otherwise, default values are given.

See 1.3.5.

1.14.7 Command: FAULT_DISPLACEMENT_LENGTH

Optional: Otherwise, default values are given.

See 1.3.6.

1.14.8 Command: FAULT_LENGTH_HEIGHT

Optional: Otherwise, default values are given.

See 1.3.7.

1.15 Simulate: Simulate sub-seismic faults

ACTION Simulate \

This module is used when parametric faults are simulated, possibly conditioned on the presence of known faults and well observations of the geological layers. The simulation is done according to the specified intensity maps and distributions for the fault properties.

1.15.1 Command: VARIO_TYPE

See 1.3.5.

1.15.2 Command: SFM_FAULT_DISPLACEMENT_LENGTH

See 1.3.6.

1.15.3 Command: SFM_FAULT_LENGTH_HEIGHT

See 1.3.7.

1.15.4 Command: INPUT_ORIGINAL_HAVANA_FAULTS

See 1.4.7

1.15.5 Command: OUTPUT_MODIFIED_HAVANA_FAULTS

See 1.4.8

1.15.6 Command: OUTPUT_HAVANA_FAULTS

Necessary

Description: Specifies a directory for output of the deterministic and simulated faults. The faults are all written on the HAVANA format used by HAVANA 5. The output directory will only contain the simulated elliptic faults.

Arguments: One. The name of the directory where the faults are to be written.

Example:
```
OUTPUT_HAVANA_FAULTS outhfdir \
```

1.15.7 Command: FAULTS_STATISTICS

Optional

Description: The most important data for the simulated faults are output to a file on an easy-to-read format.

Arguments: One, two or three. First, the name of the output file for the statistics. The format for this file is presented in Section A.4. The length of the fault is not the major diagonal of the elliptic plane, but rather the part of this diagonal that is not truncated away by other faults. Similarily for the height values.

To obtain better compatibility with the format for inputing Elliptic faults, one may add the word “NoTruncations” as the second argument of this command. Then, the untruncated lengths and heights will be output. The option “TruncInfo” will output the number of faults which truncates the given one, and their fault names.

Fault statistics file written with the “NoTruncations” and “TruncInfo” may be imported in RMS.

Examples:

```
FAULTS_STATISTICS statistics.dat \
FAULTS_STATISTICS statistics.dat NoTruncations \

FAULTS_STATISTICS statistics.dat TruncInfo \

FAULTS_STATISTICS statistics.dat NoTruncations TruncInfo \
```

1.15.8 Command: SIMULATION_VOLUME

Necessary

Description: Defines the boundary of the volume where the faults are simulated, represented by their centerpoints.

Arguments: One of two forms:

- The arguments are the names of the top and bottom horizons of the reservoir for which faults are to be simulated. Laterally, the centerpoints of faults will then only be placed where both horizons exist and have non-missing values. Vertically, centerpoints of faults may be placed between the horizons, but also in a buffer below and above the horizons. The size of this buffer is such that all faults will normally intersect the reservoir. However, the size of the buffer may also be specified by the user, using the keyword VERTICAL_BUFFER_SIZE, see the next section.
- The command has six numbers as arguments. These numbers specify the range of the centerpoints of generated faults in the following manner: minimum and maximum for the x-coordinate, and then for the y- and z-coordinates.

Examples:

```
SIMULATION_VOLUME
      ../irapsurfaces/munin_top.igri
      ../irapsurfaces/munin_bot.igri \

SIMULATION_VOLUME
   457000 466000 6574000 6587000 2100 3500 \
```

1.15.9 Command: VERTICAL_BUFFER_SIZE

Optional

Description: Defines the size of the vertical buffer above and below the reservoir used when simulating faults; see the SIMULATION_VOLUME command. If the VERTICAL_BUFFER_SIZE command is not used, the program computes a suitable buffer size.

Arguments: One. The size of the buffer.

Examples:

```
VERTICAL_BUFFER_SIZE 0 \

VERTICAL_BUFFER_SIZE 50 \
```

1.15.10 Command: INPUT_FAULT_CENTER_LINES

Optional

Description: Fault center lines are used to simulate (elliptic) new faults. The fault center lines are read from a ASCII file.

File format for the ASCII fault center lines file:

```
n               ! Number of faults to be simulated from fault center lines
missing value   ! The definition of the missing value
Faultname_1     ! name 1 of n
normal_1        ! Normal fault 1, reverse fault 0
n_1             ! Number of fault center lines points for this fault
x_11 y_11 z_11  ! Point 1 of n_1 points
dipD_11         ! Dip direction, 1 for east, 0 for west
dipA_11         ! Dip angle, 0 to 90 degrees
throw_11        ! Local throw
x_12 y_12 z_12  ! Point 2 of n_1 points
dipD_12
dipA_12
throw_12
   :
Faultname_2     ! name 2 of n
normal_2
n_2
x_21 y_21 z_21  ! Point 1 of n_2 points
dipD_21
dipA_21
throw_21
   :
```

Note: The values for normal/reverse fault, dip direction, dip angle and local throw can be given as missing values if they are not known. The values will then be drawn from the stochastic model. The choice of missing value is defined in the fault center lines file.

Arguments: The name of a file containing the fault center lines.

Example:

```
INPUT_FAULT_CENTER_LINES
   faultcenterlines.dat \
```

1.15.11 Command: FAULT_CENTER_LINES_OPTION

Optional

Description: Specifies the method used in INPUT_FAULT_CENTER_LINES for generating faults from fault center lines. The only legal values are 0 and 1. The algorithm used for option 1 is using the two endpoints of the fault center line to define the length, strike and location of the centre. The displacement is found as the maximum observed displacement. The dip is estimated from the observed dips. The algorithm used for option 0 is using all the points for estimating the length, strike and location of the centre. The default is to use algorithm 0.

Example: `FAULT_CENTER_LINES_OPTION 1 \`

1.15.12 Command: INPUT_WELL_PATHS

Optional:

Description: This command is used to specify the wells. For each well, the name of the well must be given, together with the name of a file specifying the well path. The format for this file is given in Section A.6.1.

Arguments: A list of triples of arguments. Each triple consists of the name of the well, the name of the file containing the well path specification, and the height of the kelly bushing. With three elements per well, the depth measurements in the z-coordinate then refer to depth below the kelly bushing. If only two arguments are found, the depth measurements are assumed to be below mean sea level (MSL) and not below the kelly bushing (KB).

Examples:

```
              INPUT_WELL_PATHS
                    A1 wellpathA1.dat 100
                    A2 wellpathA2.dat 100 \

      INPUT_WELL_PATHS
                    A4 wellpathA1.dat
                    A5 wellpathA2.dat  \
```

1.15.13 Command: INPUT_WELLOBS_OF_FAULTS

Optional

Description: Specifies a collection of points in the reservoir where a fault has been observed. The points are read from an ASCII file. For each point, one may also specify an interval for the throw, strike and dip of the fault at the point, and one may specify whether it is normal, or whether it is dipping eastwards. Missing information may be replaced with a '?' in the file. See Section A.6.2 for a precise specification of the format.

Arguments: One. The name of a file containing the well fault observations.

Example:

```
INPUT_WELLOBS_OF_FAULTS   wellobs.dat \
```

1.15.14 Command: INPUT_WELLOBS_OF_NOFAULTS

Optional

Description: Used to put restrictions on faults intersecting the well paths. For given intervals along the well paths, one may specify that no fault intersecting the well in this interval has displacement (at the intersection point) above a certain threshold. The intervals and thresholds are read from a file; see Section A.6.4 for the format of this file.

Arguments: One. The name of a file containing the intervals and thresholds.

Example:

```
INPUT_WELLOBS_OF_NOFAULTS   welldata.dat \
```

1.15.15 Command: NUMBER_OF_FAULTS

Necessary

Description: Determines the number of new faults to be simulated. Note that if deterministic faults, these are additional.

Arguments: One. A positive integer.

Example:

```
NUMBER_OF_FAULTS 100 \
```

1.15.16 Command: RELATIVE_INTENSITY

Optional

Description: Specifies a trend function for the relative intensity of faults. The term intensity is defined as the expected number of events (i.e. fault center points) per unit area. HAVANA normalizes the values in the intensity field, so that multiplying all the values in the trend maps with a fixed constant C will not change the result.

This command is an alternative to command DISPLACEMENT_INTENSITY in Section 1.15.18.

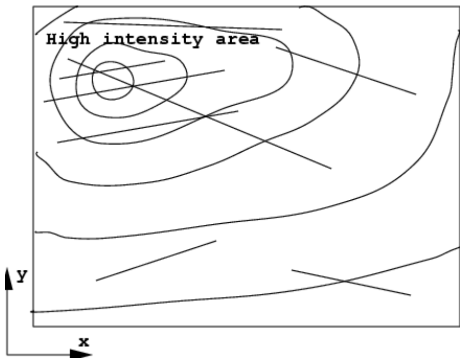


Figure 1.1. Relative intensity and simulated faults.

There are different possible ways of specifying spatially varying trend functions:

1. Using the **Constant** keyword:
   The argument is one real number.
   This trend function is a constant position independent value.
2. Using the **MultiSurface** keyword:
   The arguments are $2N$ file names. The first N files are 2D maps containing a depth surface (TVD). The last N files are 2D maps containing a value of the variable this trend function represents. All files represent grids which must cover exactly the same area and have the same grid resolution. The depth surfaces must be specified in sorted order with the most shallow surface first and the deepest one as the last one. The surfaces should not intersect each other to ensure the same order in all points (x, y).

   The procedure for defining the value of the trend function at at position (x, y, z) is as follows:

   - Find the grid cell index corresponding to the position (x, y).
   - If the z coordinate is between two depth surfaces, the value will be the linear interpolation of the values of the two grids, interpolating along the vertical line through the point.
   - If the z coordinate is above the top or below the bottom depth surfaces, the value in the first or last value file in position (x, y) is assigned.

   As one can see from this procedure, a 3D trend is defined from values of the trend function located at N different surfaces in space.

Arguments: One. A trend function. Note that the relative intensity must be non-negative, with some positive values.

Examples:

```
RELATIVE_INTENSITY
   Constant 1.0 \

RELATIVE_INTENSITY
   MultiSurface
      munin_top.s
      munin_bot.s
      munin_top_intensity.s
      munin_bot_intensity.s \
```

1.15.17 Command: RELATIVE_INTENSITY_GRID

Optional Only used if command RELATIVE_INTENSITY is specified.

Description: Specifies number of simulation box gridcells nx, ny, nz in x-, y- and z- direction respectively for the relative intensity grid. The default numbers are: $nx = 50$, $ny = 50$ and $nz = 10$.

Arguments: Tree. Integers.

Example: `RELATIVE_INTENSITY_GRID  100 100 1 \`

1.15.18 Command: DISPLACEMENT_INTENSITY

Optional

Description: Specifies a trend function for the displacement intensity of simulated mother faults. This command is an alternative to the commands RELATIVE_INTENSITY (Section 1.15.16) and REPULSION (Section 1.15.30).

Arguments: One. A **MultiSurface** trend function. For a description of its format, see Section 1.15.16.

Example:

```
DISPLACEMENT_INTENSITY
   MultiSurface
      munin_top.s
      munin_bot.s
      munin_displ_intensity.s
      munin_displ_intensity.s \
```

1.15.19 Command: DISPLACEMENT_INTENSITY_GRID

Optional Only used if command DISPLACEMENT_INTENSITY is specified.

Description: Specifies number of simulation box gridcells nx, ny, nz in x-, y- and z- direction respectively for the displacement intensity grid. The default numbers are: $nx = 50$, $ny = 50$ and $nz = 1$. Note that the number of gridcells will influence the execution time of the Metropolis algorithm heavily.

Arguments: Tree. Integers.

Example: `DISPLACEMENT_INTENSITY_GRID  100 100 1 \`

1.15.20 Command: OUTPUT_DISPLACEMENT_INTENSITY

Optional

Description: When the DISPLACEMENT_INTENSITY command is used, one may use the OUTPUT_DISPLACEMENT_INTENSITY command to output the displacement intensity of the realization produced by the simulation. One may also output the target displacement intensity, for comparison. This target intensity is computed by the program from the input in the DISPLACEMENT_INTENSITY command. Both intensities are output on a simple grid format:

```
nx ny nz
for (i=0; i< nx*ny*nz; i++)
  grid[i]
```

This is the grid used internally in the program when it is trying to match the target intensity.

Arguments: One or two. The first argument is the name of the file where the result intensity will be written out. If there is a second argument, it should also be a file name, and the target intensity will be written out there.

Examples:

```
OUTPUT_DISPLACEMENT_INTENSITY simDisplIntensity.dat \
OUTPUT_DISPLACEMENT_INTENSITY simDisplIntensity.dat
                              targetIntensity.dat \
```

1.15.21 Command: DISPLACEMENT_INTENSITY_PARAMETERS

Optional

Description: One may use this command to change from their default settings some of the parameters used in the displacement intensity simulation. Specifically, the first argument is the number of blocks (in the displacement intensity grid) used when smoothing the displacement intensity before matching it with the target density. The default value is 0. The second (and optional) argument is the constant used in the error estimation of the simulation. The default value is 0.00000001. A larger value will give realizations which match the target density less well, but the convergence of the iteration will be faster. A smaller (but positive) argument will make the program try harder to match the exact target density, but the convergence will be slower.

Arguments: One or two. The first is the number of grid cells used in smoothing, while the second is the constant used in the error computations when matching a simulated displacement density with the target density.

Examples:

```
DISPLACEMENT_INTENSITY_PARAMETERS 4 \
DISPLACEMENT_INTENSITY_PARAMETERS 0 10 \
```

1.15.22 Command: DISPLACEMENT

Necessary

Description: Specifies parameters for the distribution of the maximal fault displacements. The displacement of a fault is illustrated in Figure 1.2, and the distribution of these follow the a truncated probability distribution like the one in Figure 1.3.

Arguments: There are two required and one optional sub commands within this command: **Range** and **FractalDimension** (required), and **Asymmetry** (optional).

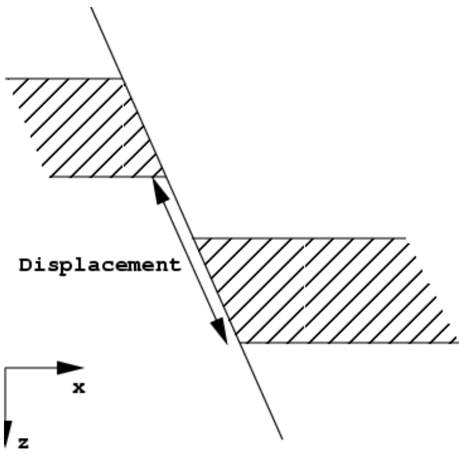


Figure 1.2. Illustration of measuring the displacement of a fault.

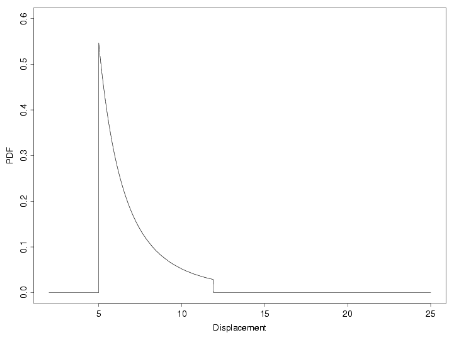


Figure 1.3. A truncated fractal probability density function, with fractal dimension $d = 2.4$ and range from 5 to 12 meters.

The arguments following **Range** are two decimal numbers. The first one is minimum displacement and the last one is maximum displacement.

The argument following **FractalDimension** is a decimal number determining the fractal dimension of the distribution.

The arguments following **Asymmetry** are two decimal numbers. The first number specifies how much of the displacement takes place on the footwall side, and how much on the hangingwall side. If it is 1, all displacement takes place on the hangingwall side; if it is zero (the default), there is equally much displacement on either side, and if it is -1, all displacement takes place on the footwall side. The second number specifies the uncertainty around the first number. If it is greater than zero, the asymmetry number will be drawn for each fault, from a normal distribution with expectation and standard deviation given by the two numbers. Note that if the **Asymmetry** sub-command does not appear, displacement will always be equally divided between the footwall and hangingwall sides.

Examples:

```
DISPLACEMENT
   Range 7.5 30
   FractalDimension 2.0 \
```

This example generates slumps:

```
DISPLACEMENT
   Range 10.0 30
   FractalDimension 2.0
   Asymmetry 1.0 \
```

1.15.23 Command: FAULT_DISPLACEMENT_LENGTH

Necessary

Description: Specifies parameters for the relationship between (maximum) fault displacement and (maximum) fault length. The fault length l is assumed to approximately be a function of the displacement d. The relationship is as follows: $l \approx (d/c_1)^{1/p}$ The uncertainty in this relationship is modeled by multiplying the right hand side in the equation above by a stochastic variable with lognormal distribution. The fault length is then $l = (d/c_1)^{1/p} V_1$ where V_1 has a lognormal distribution, so that $\log_e(V_1)$ has a normal distribution with expectation zero and standard deviation σ_1.

Arguments: Three. The exponent p, the constant c_1 and the standard deviation σ_1, all real numbers.

Values of σ_1 close to 0 (0.1 - 0.2) indicate small uncertainty while larger values (0.5 - 1) indicate more uncertainty in the relationship.

Example: `FAULT_DISPLACEMENT_LENGTH 1.10 0.01 0.05 \`

1.15.24 Command: FAULT_LENGTH_HEIGHT

Necessary

Description: Specifies parameters for the relationship between (maximum) fault length and (maximum) fault height. The fault height h is illustrated in Figure 1.4, and is approximately following the relationship $h \approx l/c_2$ as a "function" of the fault length l. The uncertainty in this relationship is modeled by multiplying the right-hand side in the equation above by a stochastic variable with lognormal distribution. The fault height is then $h = (l/c_2)V_2$ where V_2 has a lognormal distribution, so that $\log_e(V_2)$ has a normal distribution with expectation zero and standard deviation σ_2.

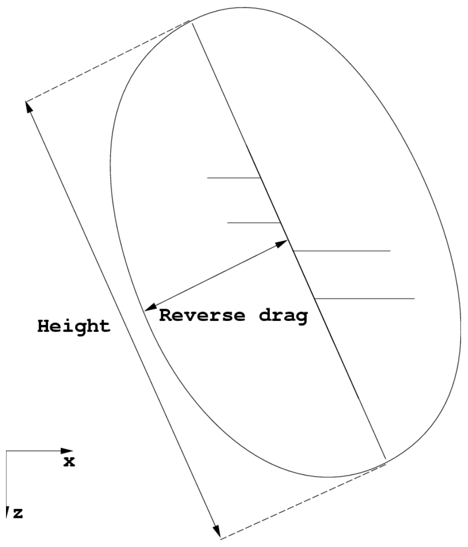


Figure 1.4. Height and reverse drag of a fault.

Arguments: Two. The parameter c_2 and the standard deviation σ_2.

Values of σ_2 close to 0 (0.1 - 0.2) indicate small uncertainty while larger values (0.5 - 1) indicate more uncertainty in the relationship.

Example:
```
FAULT_LENGTH_HEIGHT 2.0 0.1 \
```

1.15.25 Command: FAULT_AVERAGE_REVERSEDRAG

Necessary

Description: Specifies parameters for the relationship between the average size of the fault plane and the (maximum) reverse drag of the fault. The reverse drag r (see Figure 1.4) is assumed to approximately follow the relationship $r \approx c_3\sqrt{lh}$ as a function of the fault height h and the fault length l. The uncertainty in this relationship is modeled by multiplying the right hand side in the equation above by a stochastic variable with lognormal distribution. The reverse drag is then $r = c_3\sqrt{lh}V_3$ where V_3 has a lognormal distribution, so that $\log_e(V_3)$ has a normal distribution with expectation zero and standard deviation σ_3.

Arguments: Two. The parameter c_3 and the standard deviation σ_3.

Values of σ_3 close to 0 (0.1–0.2) indicate small uncertainty while larger values (0.5–1) indicate more uncertainty in the relationship.

Example:
```
FAULT_AVERAGE_REVERSEDRAG 0.40 0.1 \
```

1.15.26 Command: ORIENTATION_GROUPS

Optional

Description: The strike, dip, and "dip down east" parameters of a fault are collectively described as the "orientation" of the fault in this manual. One may specify several distinct groups of faults and then control the orientation of the faults in each group separately. In each group, the orientation may in fact vary across the reservoir.

To use more than one group of faults in this sense, one must use the command ORIENTATION_GROUPS. It specifies the probability for *mother faults* to belong to the different groups. Orientation parameters for each of the groups must be specified in the STRIKE and DIP commands.

Arguments: Positive decimal numbers specifying the probability of each of the orientation groups. HAVANA normalizes the specified values to probabilities. The number of values will give the number of orientation groups.

Examples:
```
ORIENTATION_GROUPS  0.3 0.7 \
```

1.15.27 Command: STRIKE

Necessary

Description: Specifies the probability distribution with related parameters for the strike. The strike of a fault is the angle between its intersection line with a horizontal plane and north (i.e., the y-coordinate direction), see Figure 1.5. The angle is measured in degrees, between 0 and 180, clockwise.

This command accepts one or more parameter sets. If the command ORIENTATION_GROUPS is used, one set for each orientation group should be specified, otherwise only one set. If ORIENTATION_GROUPS is specified and only one parameters set is specified, the same parameters will be used for all groups. The sets of numbers for

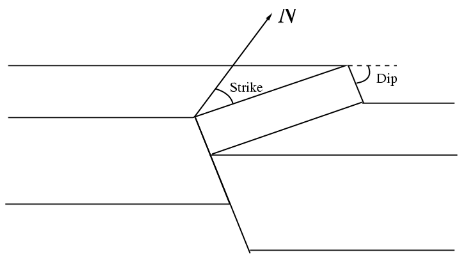


Figure 1.5. Measurement of strike and dip.

each group must be separated with a ' ; ' character (Note the blank space before and after). The probability distributions are specified by the keyword **Gaussian**. Additional truncation limits can be specified by the subcommand **Limits**.

Arguments: Each parameter set consists of one or two sub commands as described below.

First the sub command **Limits** with two parameters. The first one is the minimum strike value and the second is the maximum strike value. This sub command is optional.

Then follows the sub command **Gaussian**. A Gaussian distribution with in general spatially varying trend functions giving the expectation and standard deviation. The expectation is specified by sub command **Expectation** followed by a trend function. The standard deviation is specified by the sub command **Stdev** followed by a trend function. See Section 1.15.16 for the available trend functions and their format. The trend functions may be used to get different strike situations in different parts of the reservoir.

Example:

```
STRIKE
   Limits 25 35
   Gaussian
      Expectation
         Constant 30
      Stdev
         Constant 3 ;

   Gaussian
      Expectation
         Constant 150
      Stdev
         Constant 3 ;
 \
```

1.15.28 Command: DIP

Necessary

Description: Specifies the distribution of the dip angle for the faults. The dip of a fault is the inclination angle between the fault plane and the horizontal, see Figure 1.5. The dip is given in degrees, between 0 and 90.

This command accepts one or more parameter sets. If command ORIENTATION_GROUPS is specified one set for each orientation group can be specified, otherwise only one set. If ORIENTATION_GROUPS is specified and only one parameters set is specified, the same parameters will be used for all groups. The sets of numbers for each group must be separated with a ' ; ' character (Note the blank space before and after).

Arguments: Each parameter set consists of four sub commands **ProbDownEast**, **ProbNormal**, **Expectation** and **Stdev**.

The sub command **ProbDownEast** is followed by a decimal number which is the probability for the fault plane dipping down towards the east. The sub command **ProbNormal** is followed by the probability of having a normal fault (contrary to a reverse fault).

Then, the expectation and standard deviation of the Gaussian distribution of the fault dip angle is specified by the sub commands **Expectation** and **Stdev**, each followed by a trend function, see section 1.15.16. These trend functions may be used to let the dip vary across the reservoir.

Example:

```
DIP
   ProbDownEast 0.7
   ProbNormal 1.0
   Expectation
      Constant 60.0
   Stdev
      Constant 4.0 ;

   ProbDownEast 0.5
   ProbNormal 1.0
   Expectation
      Constant 40.0
   Stdev
      Constant 4.0 ;
\
```

1.15.29 Command: FAULT_TRUNCATION

Necessary

Description: Specifies a parameter controlling when one fault should truncate another. When two fault planes intersect, it is always the fault appearing first in the ordered list of faults that may or may not truncate the other one; the second fault can never truncate the first.

If the given parameter is greater than or equal to zero, and less than or equal to one, then truncation will be decided according to the following rule: The length of the line of intersection between the fault planes is compared with the length of the extension of this line in the latter fault of the list. If the ratio between these lengths is above the given parameter, then truncation occurs; otherwise, it does not.

If the given parameter is less than zero, then truncation is decided stochastically: The relative intersection fraction is computed as above, and is is then used as the probability for intersection.

Arguments: One. A decimal parameter, the truncation limit. Values close to 1 indicate very little truncation, while values closer to 0 indicate truncation of all faults intersecting. Negative numbers indicate stochastic truncation.

Example: `FAULT_TRUNCATION 0.01 \`

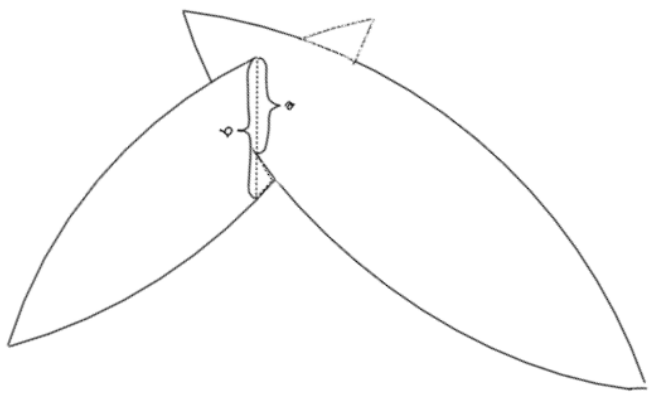


Figure 1.6. Relative intersection determining the truncation of faults.

1.15.30 Command: REPULSION

Optional

Description: Specifies parameters regarding the spatial interaction (repulsion) between parent faults. This command is an alternative to command DISPLACEMENT_INTENSITY in Section 1.15.18.

Arguments: Two. The first argument is the interaction range, a decimal number. If two parent faults are further apart than the interaction range, they will not repel each other. The second argument is the maximum negative interaction potential, indicating the strength of the interaction. If the absolute value of this parameter is large, the repulsion is strong, if it is close to zero, the repulsion is weak.

Example: `REPULSION 2000.00  -1.00 \`

1.15.31 Command: DISPLACEMENT_WEIGHT

Optional

Description: The parameter specified here is an exponent for the fault displacements when these are used to determine family members for the mother faults. The probability for the ith mother with displacement d_i to be selected as the mother for a new fault is originally $d_i/\sum d_i$. This implies that very large faults easily will become mothers for too many children faults. To decrease the impact of the displacement size a weight is introduced. The weighting function is d_i^p, where d_i is the displacement and p is the exponent parameter given here. The new probabilities are $d_i^p/\sum d_i^p$. Giving values larger then 1 increase the significance of the displacement in this relationship, while values smaller than 1 gives more

Arguments: One. A decimal number for the exponent p. Default is $p = 1$.

1.15.32 Command: NUMBER_OF_FAMILIES

Necessary

Description: Specifies the expected number of families to be used. This number should be the desired expected sum of the seismic faults, the mother faults in earlier simulated fault sets and the new simulated mother faults.

The actual number of families is drawn from a binominal distribution with the given number as the expectiation value.

Setting the expected number of families to 1 will usually prevent simulation of new mother faults, given that the number of input mother faults are sufficiently large.

Arguments: One. An integer.

Example: `NUMBER_OF_FAMILIES 10 \`

1.15.33 Command: CHILDREN_PARAMETERS

Necessary: if children faults are to be simulated and neither CHILDREN_PARAMETERS_GENERAL nor RELAY_RAMPS is given.

Description: Specifies how the center points of the children faults are distributed, relatively to the mother faults. These points are placed using a multinormal distribution around the center point of the mother fault.

Arguments: Three. The standard deviations along the length of the fault, along the height of the fault, and normal to the fault plane, respectively. All these numbers are relative to the dimensions of the current mother fault. For example, using the parameters 0.5 0.5 0.5 means that almost all children faults will be placed inside the mother fault ellipsoid. Increasing the first number means that children may appear further away along the length of the mother fault. Increasing the last number means that children may appear further away from the mother in the direction normal to the mother fault plane.

Example: `CHILDREN_PARAMETERS 0.9 0.3 0.7 \`

1.15.34 Command: CHILDREN_STRIKE

Necessary: if children faults are to be simulated and neither CHILDREN_PARAMETERS_GENERAL nor RELAY_RAMPS is given.

Description: The fault strike for the children faults is assumed to follow a normal distribution where the strike of the mother fault is the expected value. This command specifies the standard deviation in this distribution, in degrees.

Arguments: One. The standard deviation of the child fault strike, a decimal number.

Example: `CHILDREN_STRIKE 10.0 \`

1.15.35 Command: CHILDREN_DIP

Necessary: if children faults are to be simulated and neither CHILDREN_PARAMETERS_GENERAL nor RELAY_RAMPS is given.

Description: The fault dip for the child faults is assumed to follow a normal distribution, where the dip of the mother fault is the expected value. This command specifies the standard deviation in the distribution.

Arguments: One. The standard deviation of the child fault dip, a decimal number.

Example: `CHILDREN_DIP 2.5 \`

1.15.36 Command: CHILDREN_PARAMETERS_GENERAL

Optional

Description: Specifies how children faults are simulated, relatively to the mother faults. This command is an alternative to the CHILDREN_PARAMETERS, CHILDREN_DIP and CHILDREN_STRIKE commands.

Arguments: The command has 14 or more arguments specifying fault names and the 14 parameters controlling the simulation of children faults.

The children faults are distributed around a single point. The first three arguments specify the position of this point relative to the centerpoint of the mother fault. The arguments are the shift in the strike, dip and reverse drag direction. The values should be relative to of the size of the mother fault (length, height and reverse drag). For example, if 1 0 0 is used, the centerpoints of the children faults will be distributed around one of the fault tips.

The distribution for the children fault centerpoints is a combination of two multinormal distributions, one in each direction along the length of the mother faults. Argument four and five specifies the standard deviations along the length of the mother fault in both directions, the next two arguments are the standard deviation along the height, and normal to the fault plane, respectively. Again, all these numbers are relative to the size of the current mother fault.

For example, using the parameters 0.5 0.5 0.5 0.5 means that almost all children faults will be placed inside an ellipsoid equal in size to the mother fault ellipsoid. Increasing the first number means that children may appear further away along the length of the mother fault. Increasing the last number means that children may appear further away from the mother in the direction normal to the mother fault plane.

The intensity of children fault centerpoints are expressed by $exp(-x^a)$, where x is the distance from the centerpoint scaled by the standard deviations. The a can be set using argument 8. For a multinormal distribution a value of 2.0 must be used.

Argument 9 controls the children fault intensity on the hangwall v.s footwall side of the mother fault. The values accepted are from -1 to 1. A value of 1 will cause zero intensity at the footwall side and double intensity at the hangwall side. A value og 0.5 will cause at 50% increased/decreased intensity at the footwall/hangwall sides. A value of 0 will cause no shift, i.e., same intensity on both sides.

The fault strike and dip for the children faults is assumed to follow normal distributions. Argument 10 is the expected dip direction, relative to the dip direction of the mother fault. Argument 11 is the dip standard deviation. The values should be given degrees.

Argument 12 and 13 is similar to 10 and 11, but for strike.

It is possible to specify several parameter sets separated by ;. Argument 14 is a number specifying the fraction of children to be simulated using the specified set of parameters. The "numbers" will be scaled if their sum is different form one.

Parameter sets may be given for specific mother faults by entering a list of fault names starting from argument 15. Faults specified will only use parameter sets where it is specified.

Note that at least one parameter set must be *without* faultnames to account for unspecified mother faults. Due to the stochastic nature of the Simulate action there is always a possibility of a new "unknown" mother fault.

Example:

```
CHILDREN_PARAMETERS_GENERAL
   1.0 0.0 0.0      ! center for intensity strike/dip/normal dir.
   0.1 1.0 0.1 0.05 ! length, length2, height, width
   2.0              ! A in exp(1/A)
   0.0              ! HangwallFrac
   0.0 0.5          ! Dip: exp. value rel. to mother and st.dev.
 -30.0 1.0          ! Strike: exp. value rel. to mother and st.dev.
   0.6              ! fraction of children
;
 ! A parameter set using a compact input style:
 ! Co   Cd   Cn   L1   L2   H    W    A    HW   Ed   Sd   Es   Ss   F
   1.0  0.0  0.0  0.1  1.0  0.1  0.05 2.0  0.0  0.0  0.5  90.0 1.0  0.4
;
   1.0 0.0 0.0      ! center offset strike/dip/normal
   0.1 1.0 0.1 0.05 ! length, length2, height, width
   2.0              ! childrenA
   0.5              ! childrenHangwallFrac
   0.0 0.5          ! Dip: exp. value rel. to mother and st.dev.
  90.0 1.0          ! Strike: exp. value rel. to mother and st.dev.
   1.0              ! fraction of children
  HF1 HF2 HF3       ! list of fault names for this param. set
/
```

1.15.37 Command: CHILDREN_PARAMETERS_MOTHER_TYPE
Optional

Description: Specifies how the center points of the children faults are distributed, relatively to the mother faults, if the mother fault is of type RMS. There are two different possibilities. The RMS fault is approximated by an elliptical fault. This method is very fast, but not very accurate. The other possibility is to use the triangle structure of the RMS. This method is accurate, but very slow, depending on the density of the triangularization and the size of the intensity field given in RELATIVE_INTENSITY_GRID or DISPLACEMENT_INTENSITY_GRID.

Arguments: One. The two options are elliptic or rms. The default value is elliptic.

Example:

```
CHILDREN_PARAMETERS_MOTHER_TYPE rms \
CHILDREN_PARAMETERS_MOTHER_TYPE elliptic \
```

1.15.38 Command: RELAY_RAMPS
Optional

Description: Specifies the relay ramp intensity between two mother faults. The intensity field is given as an area between parts of the two mother faults, with a planar top and bottom boundary. The children faults are distributed according to the given parameters.

Arguments: The first two parameters specifies the names of the interacting mother faults. The next two parameters specify the intensity field between two mother faults as a function of the fractional length of each mother fault, that is, the part of the faults to be included in the relay ramp field. Four parameters describe the intensity field between the interacting mothers. The two first numbers represent the size of the intensity along the strike of the two mothers. The intensity field varies acording to a linear function between the mother faults, and a breakline, dividing the intensity field in two different parts. The third number describes the relative distance from the first mother fault to the breakline. A number 0.5 means that the breakline divides the intensity field between the two mothers in half. The fourth number the size of the intensity at the break line. Please note that the intensity sizes are relative, that is, the intensity field given by 1.0 1.0 0.5 0.5 and the field 2.0 2.0 0.5 1.0 are both equal. The next number is the likelyhood of having a connecting fault between the two mother faults. A number 1.0 means that there will be a connecting fault, and a number 0.0 means that there will be no connecting fault. The two parameters that follow the likelyhood of a connecting fault, specifies the minimum and maximum displacement of the connecting fault. If the maximum value of the displacement is set too low, it may be impossible to draw a connecting fault. The next two numbers gives the standard deviation of strike and dip for the drawn children. Note that the expected strike is parallel to the axis of the relay ramp as defined between the two mother faults. The last parameter is an number specifying the fraction of children, for the two mother faults, to be used for the relay ramp simulation. For example, if the relay ramp structure has a fraction of 0.3 and the children parameter general structure has a fraction of 0.7, this means that 70% of the simulated children, belonging to the two mother faults, will be simulated from the children parameter general family of children faults. The remaining 30% is simulated from the relay ramp family of children faults. It is possible to specify several relay ramps, separated with ;.

Example:

```
RELAY_RAMPS
HF1 HF2             ! mother faults
0.5 0.5             ! relative length of intensity field along mother faults
1.0 1.0 0.5 0.5     ! intensity field between mothers
0.5 3 5             ! connecting fault between mothers
10 10               ! strike and dip standard devations
0.3 ;               ! fraction of children
HF3 HF4             ! mother faults
0.3 0.2             ! relative length of intensity field along mother faults
1.0 0.5 0.25 0.1    ! intensity field between mothers
0.8 2 4             ! connecting fault between mothers
15 10               ! strike and dip standard devations
0.5 \
```

1.15.39 Command: NUMBER_OF_ITERATIONS

Optional

Description: Specifies the number of iterations to be used in the simulation procedure for the mother faults.

Arguments: One. An integer.

Example: `NUMBER_OF_ITERATIONS 50000 \`

1.15.40 Command: FAULTNAME_PREFIX

Optional

Description: Specifies a prefix for the fault names for the generated faults. Defaults to 'HF'.

Arguments: One. A string.

Example: `FAULTNAME_PREFIX MyFaults \`

1.16 CreateFaultGrid: Create fault zone grid

ACTION CreateFaultGrid \
The CreateFaultGrid action is used to create local grid refinement used for 3D modelling of the fault zone. It is both possible to generate a grid that only consists of the fault zone grid or a grid consisting both of the coarse background grid and the locally refined fault zone grid.

The commands supported by the CreateFaultGrid action are described below:

1.16.1 Command: INPUT_ECLIPSE

Required

Description: The ECLIPSE grid that is basis for the fault zone grid. This grid must contain the fault trace data (ECLIPSE `FAULTS` keyword), and fault block identifiers given by a grid parameter named `FaultBlock`. This parameter can be created in RMS from the `Grid index parameters...` job under `Parameter utilities` in the context menu for the grid.

Arguments: One or more names of ECLIPSE files.

Example:

```
INPUT_ECLIPSE grid_file.grdecl poro.grdecl perm.grdecl \
```

1.16.2 Command: OUTPUT_ECLIPSE

Optional

Description: Output ECLIPSE file containing the coarse grid given by INPUT_ECLIPSE and the fault zone grid given as a local grid refinement.

Arguments: One or more file containing listed output keywords. If no keywords are listed for the last file, all remaining keywords are written to this file. Each file is separated by a semi-colon.

Examples:

```
OUTPUT_ECLIPSE eclipseout.grdecl \

OUTPUT_ECLIPSE
       SPECGRID  COORD   ZCORN  ACTNUM    grid.grdecl ;
       PERMX     PERMY   PERMZ            perm.grdecl ;
       CARFIN    ACTNUM                   localgrids.grdecl ;
       remain.grdecl
\
```

1.16.3 Command: EXPORT_LOCAL_GRIDS

Optional

Description: Output ECLIPSE file only containing the finescaled faultzone grid.

Arguments: The arguments and syntax for this keyword exactly the same as for OUTPUT_ECLIPSE, 1.16.2.

Examples:

```
EXPORT_LOCAL_GRIDS faultzone.grdecl \

EXPORT_LOCAL_GRIDS
       SPECGRID  COORD   ZCORN  ACTNUM    grid.grdecl ;
       PERMX     PERMY   PERMZ            perm.grdecl ;
```

```
                    CARFIN    ACTNUM                    localgrids.grdecl ;
                    remain.grdecl
             \
```

1.16.4 Command: GRID_REFINEMENT

Required

Description: The refinement factors used when generating the local grid refinements.

Arguments: 3 arguments:

1. The refinement factor in x direction.
2. The refinement factor in y direction.
3. The refinement factor in z direction.

Example:

```
GRID_REFINEMENT 3 3 2 \
```

1.16.5 Command: REFINEMENT_WIDTH

Required

Description: How many cells on each side of the fault trace that are refined.

Argument: Number of cells.

Example:

```
REFINEMENT_WIDTH 2 \
```

1.17 MergeFaultGrid: Merge fault zone grid with coarse grid

ACTION MergeFaultGrid \
The MergeFaultGrid action is used to merge a fine-scale faultzone grid with a coarse-scale host grid. The faultzone grid must be created using the CreateFaultGrid action. The faultzone grid is added to the coarse grid as a set of local grid refinements.

1.17.1 Command: INPUT_GLOBAL_GRID

Required

Description: The original coarse-scale host grid that the faultzone grid shall be merged with.

Arguments: One or more names of ECLIPSE files.

Example:
```
INPUT_GLOBAL_GRID grid_file.grdecl poro.grdecl perm.grdecl \
```

1.17.2 Command: INPUT_LOCAL_GRID

Required

Description: The fine-scaled local faultzone grid generated by the CreateFaultGrid action.

Arguments: One or more names of ECLIPSE files.

Example:
```
INPUT_LOCAL_GRID grid_file.grdecl poro.grdecl perm.grdecl \
```

1.17.3 Command: OUTPUT_ECLIPSE

Required

Description: The grid file(s) where the merged shall be written to.

Arguments: One or more file containing listed output keywords. If no keywords are listed for the last file, all remaining keywords are written to this file. Each file is separated by a semi-colon.

Examples:
```
OUTPUT_ECLIPSE eclipseout.grdecl \

OUTPUT_ECLIPSE
       SPECGRID  COORD   ZCORN  ACTNUM    grid.grdecl ;
       PERMX     PERMY   PERMZ            perm.grdecl ;
       CARFIN    ACTNUM                   localgrids.grdecl ;
       remain.grdecl
\
```

1.17.4 Command: LOCAL_GRID_NAME

Optional

Description: Name used for local grid refinements created from faultzone grid. Defaults to `LGR`.

Argument: Local grid refinement name.

Examples:
```
LOCAL_GRID_NAME fzg \
```

A File formats

A.1 Input of structural model exported from RMS - new format

In the "Cohiba development branch" of RMS 2011 a new export of the complete structural model was introduced. This export is available by choosing the `Extract fault data to files...` job for the desired horizon model.

The whole structural model is exported to a single folder. The definition of the structural model is given in the `fault_model` file. All fault line points are exported in a single file named `fault_lines`, while the fault surfaces for each fault is given in files names `<fault-name>_grid` and the horizons are exported in files with the same name as the horizon.

The RMS 2012 export has the same file formats as the previous RMS 2011 export, but with some additional data added and some data removed. HAVANA is compatible with the 2012 export format and register the version of the export format through the version number given at the beginning of the structural model file. The RMS 2012 export format has version number 3.0.

A.1.1 Structural model definition

The structural model file contains the following sections:

BoundingBox The definition of the used fault model bounding box.

Faults Number of faults, and name of each fault in this structural model. For each fault azimuth, dip and type of fault is given. The dip and azimuth information is also given by the transformation matrix in the fault file, and is hence not used.

Truncations The fault truncations. Note that Havana currently only supports simple truncations, i.e. only one truncation rule per line.

Horizons The names of the horizons in the structural model.

FaultBlocks The definition of the fault blocks in the model.

Example of a structural model definition file:

```
Version 2.0
#
# syntax:
# Lines starting with '#' are comment lines
# Data is separted by white space which is either ' ' or a tab
#
# This file is designed to allowed a external program create a fault model
# similar to the one used inside rms.
#
# The file is broken into three sections:
#
# The Bounding box of the model
# The faults
# The fault to fault truncations
```

```
#
# To read this file look for the keywords
# 'Version,BoundingBox,Faults,Truncations'
# Each section has a fixed format after that
#
#-----------------------------------------------------
# Keyword to the section
BoundingBox
#
# The bounding box for the fault model. The bounding box can be
# rotated arround the vertical axis. It is defined by a center
# point, the length of the three sides and anti clockwise rotation
# when viewed from the top
#
#
# Box center east north depth (must used doubles to read this offset)
463087.683838 5933652.235840 1896.718506
# Box size east north depth
7762.404297 7672.138184 866.392700
# Box rotation in degrees (clockwise viewing from the top)
0.000000
BoundingBoxEnd
#
#
#-----------------------------------------------------
# Keyword to the section
Faults
#
# First the number of faults
FaultCount 7
#
# Each fault is formed into a single valued surface form a particular
# gridding direction. The dip & dip az (in degrees) is the gridding
# direction used. The format is as follows. fault names should have
# no white space in them. The dip & dipaz define a normal. This normal
# points towards the above side of the fault.
#
# Fault-Type N: normal, R: reverse, U: undefined
#
# format: fault_name azimuth dip fault_type
F1 60.914519 57.463298 N
F2 44.124806 90.000000 N
F3 231.591778 90.000000 N
F4 213.517344 90.000000 N
F5 208.746356 90.000000 N
F6 91.633742 90.000000 N
F7 79.927519 90.000000 R
FaultsEnd
#
```

```
#
#----------------------------------------------------
# Keyword to the section
Truncations
#
# We don't really know how many truncations we might get so
# keep reading them until you find 'TruncationsEnd'. In the
# user interface:
# About = Hanging wall or HW
# Below = Footwall or FW
# keep reading them until you find 'TruncationsEnd'
#
# faultA > faultB     ...faultA is truncated above faultB
# faultA < faultB     ...faultA is truncated below faultB
# faultA < faultB & > faultC
# faultA is truncated where it is below faultB and above faultC
#
#The lists should only contain faults named in the Faults section
#
F2 > F1
F3 > F1
F3 > F2
F4 < F3
F6 < F3
#
TruncationsEnd
#
#
#----------------------------------------------------
# Keyword to the section
Horizons
#
#Just horizon names. Cannot contain spaces.
#
TopC
TopB
TopA
BaseA
#
HorizonsEnd
#
#
#----------------------------------------------------
# Keyword to the section
FaultBlocks
#
#The fault block id is an integer. We then list which side
#the fault block is of neighbor faults:
#
```

```
#Like for truncations > means HW side and < means FW side
#
1 > F1
3 < F1 > F2
8 > F5 > F6
9 < F1 > F3 > F4 < F5 > F6
10 < F1 < F2 > F3 < F4 > F6
12 > F3 < F6 > F7
13 < F6 < F7
14 < F1 < F2 < F3
#
FaultBlocksEnd
```

In the RMS 2012 export the format of the faults in the structural model file is "`fault_name fault_type age_group usage`", where the age group and usage are new parameters. The usage is given as either USED, NO_THROW, GHOST or UNUSED. The NO_THROW parameter means that the fault have no displacement. The GHOST parameter means that the fault is only there due to truncations. Further, the input parameters azimuth and dip are removed.

The horizon format is given as "`horizon_name horizon_type age_group`", where the two latter parameters are new. The horizon type gives whether the horizon is a erosion surface, i.e. UNCONFORMITY or an ordinary horizon, i.e. DEPOSITION.

Age is also given for the fault blocks, however in the form of an age interval. The fault block format is as follows "`block_id min_age max_age neighbor_faults`".

A.1.2 Fault surface file

The fault file is named `<fault-name>_grid` and has a format similar to the `fault_model` file. The file contains the following sections:

TransformMatrix4x4 A 4x4 transformation matrix defning the transformations between local and global coordinates.

NU and NV number of points in local u and v directions.

Data NU*NV values with the local w-value (distance to reference plane) for each point on the fault surface.

Thickness NU*NV values, currently only used to define the fault tip. Positive thickness values means that the point is within the fault tip, while zero or a negative value means that the point is outside the fault tip line.

Example fault surface file:

```
RMS_fault_grid_version 2.0
#
# Comment-lines start with #
#
# Definition of local space by 4x4 matrix
# Use the matrix M to transform local grid points to user coordinates
#
# [x, y, z, w] = M * [u, v, data[u][v], 1.0
#
# u,v are indexes into the data/thickness sections.
# Rotation, reference point and grid increments are embedded in the
# matrix.
#
TransformMatrix4x4
-48.6113950022 -47.0015702157 73.6735003873 467405.018948
87.3895432872 -26.1451409223 40.9816952212 5929790.22863
0 -84.30470926 -53.7839752852 2677.77285235
0 0 0 1
EndTransformMatrix4x4
#
# Number of cells in U direction
#
NU 98
#
# Number of cells in V direction
#
NV 99
#
# 2d array of grid values.  NU * NV values.
#
Data
-0.840338
-0.88161
-0.970237
-1.08047
-1.19489
    .
    .
EndData
#
# Thickness attribute.
# 2d array of grid values.  NU * NV values.
# Positive where the fault is active, negative outside the active
# area.
# The 0-contour of this attribute is the tipline of the fault.
#
AttributeThickness
-0.248483
-0.209946
-0.119374
0.0196044
0.190451
    .
    .
EndAttributeThickness
```

A.1.3 Fault line file

The fault lines for alle the fault surfaces and all the horizons are given in the `fault_lines` file. This file is on RMS internal points format, see section A.11. For each fault line point the following attributes are given:

FaultSide Side of fault that the fault line point belongs to. Either `hw` or `fw`.

Fault Name of fault that the fault line point belongs to.

Horizon Name of horizon that the fault line point belongs to.

FaultBlock ID of fault block where this fault line point is located.

Example file:

```
String   FaultSide
String   Fault
String   Horizon
Discrete FaultBlock
466914.549  5929860.689  2105.067  hw  F1  TopC  1
466758.129  5929773.680  1764.236  fw  F1  TopC  9
466865.918  5929948.068  2104.987  hw  F1  TopC  1
466705.468  5929858.816  1764.563  fw  F1  TopC  9
                     .
                     .
```

In the RMS 2012 export a new attribute "Extrapolation" is given to the fault line points. This parameter is 0 if the fault line point is eroded, and 1 if not.

A.1.4 Horizon file

The horizons are exported to files with the same name as the horizons. They are exported as point sets on the RMS internal points format, see section A.11. Two attributes are given for each point, a `FaultBlock` attribute giving witch fault block the point belongs to, and a `FaultTag` attribute that is currently not in use.

Example file:

```
Discrete  FaultBlock
String  FaultTag
459206.482  5929816.167  1730.060    13  UNDEF
459406.482  5929816.167  1731.072    13  UNDEF
459606.482  5929816.167  1732.178    13  UNDEF
459806.482  5929816.167  1733.431    13  UNDEF
                       .
                       .
```

In the RMS 2012 export the attribute "FaultTag" is removed from the horizon file. Horizon patches are exported one fault block at the time. The points are in addition to fault block number tagged with and active flag. The points form a rectangle for each fault block. The nodes that contribute to cells touching the fault block have an active flag = 1, and the points completely outside the block has active flag = 0.

A.2 Input of fault model from RMS on old format

In all other versions of RMS than the cohiba development branch of RMS2011, the only export avaiable is by generating a fault model file. This file is written when running the `fault modelling` job in RMS, and is written to a location specified by the `RMS_FAULT_MODEL_FILENAME`.

The fault surfaces must be exported on the RMS points format to the same directory containing the fault model file, and the files must have same names as fault model file.

The fault model file is similar to the fault model file exported with the new export job, see section A.1.1, however it only contains the following information:

BoundingBox The definition of the used fault model bounding box.

Faults Number of faults, and name of each fault in this structural model. For each fault azimuth and dip is given.

Truncations The fault truncations. Note that Havana currently only supports simple truncations, i.e. only one truncation rule per line.

A.3 Input of Elliptic faults

This format can be read from RMS, and Havana is able to read and write this format. The format is as follows:

The lines starting with a # are comment lines.

Each remaining line of the file contains data for one fault. Number of faults are counted from the list. The numbers read are interpreted as:

- Name of fault.
- x-coordinate of the center point of the fault.
- y-coordinate of the center point of the fault.
- z-coordinate of the center point of the fault.
- The total maximal displacement.
- The asymmetry of the displacement (A number between -1 and 1: -1 means that all displacement happens on the foot-wall side; 1 means that all displacement happens on the hanging-wall side, and 0 means a symmetric fault).
- The strike, measured clockwise from the north, in degrees.
- The dip, in degrees, such that vertical faults have dip 90.
- The total (untruncated) length of the fault.
- The total (untruncated) height of the fault plane.
- The range of the fault (distance from center point to where the fault operator dies out, measured in the direction normal to the fault plane).
- Whether the fault is normal: The input must be either 1 (if the fault is normal), or 0.

As an example, the following will be a legal input file.

```
#Total number of output Elliptic faults: 14
#FaultName        x         y           z    Displacement  Asym  Strike   Dip    Length    Height   Range     Norm

HF1          460120.71  6583039.71   2813.25     25.98  0.00   53.65   67.55  1268.99   760.03   197.72   1
HF2          463713.80  6580121.80   2941.61     23.56  0.00   57.86   84.53  1219.82   667.54   165.42   1
HF3          460057.22  6584665.58   2634.95     21.38  0.00  137.97   79.91  1087.29   671.07   159.27   1
HF4          460115.32  6581230.04   2505.64     22.61  0.00  167.44   67.92  1015.46   438.66   141.55   1
HF5          462476.17  6581641.80   2617.91     25.78  0.00  152.40   75.83  1307.16   662.10   181.02   1
HF6          462160.33  6583824.32   2993.71     26.77  0.00  154.53   85.95  1331.55   668.31   214.93   1
HF7          463932.37  6581564.19   2984.36     22.66  0.00  151.45   76.84  1098.69   610.64   196.89   1
HF8          463211.27  6584549.08   2707.61     24.34  0.00  155.60   88.43  1220.21   642.74   194.54   1
HF9          460863.05  6581489.07   2656.46     19.56  0.00  155.97   75.53   956.30   512.86   126.46   1
```

```
HF10          464787.89  6582171.35   2728.76     20.05  0.00  147.89   77.78  1016.95   478.11   176.26    1
HF11          462628.23  6580536.22   2993.08     22.50  0.00   61.82   78.77  1116.67   693.65   163.70    1
HF12          463722.50  6581516.10   2619.67     23.37  0.00  155.87   82.07  1132.20   548.95   160.55    1
HF13          461410.30  6581764.77   2553.99     19.91  0.00  154.21   72.05   969.70   470.18   141.55    1
HF14          464945.67  6581687.74   2639.33     22.69  0.00  162.62   80.09  1054.77   473.31   148.67    1
```

A.4 Output of fault statistics

The format for the output of fault statistics using the keyword FAULTS_STATISTICS (see Section 1.15.7) is very similar to the format for inputing Elliptic faults (see Section A.3). The only differences are that in the faults statistics file, the *truncated* lengths and heights of the faults will normally be output, and that the indices indicating fault truncations that may appear at the end of each line in the format for inputing Elliptic faults do not appear in the statistics format.

A.5 Fault center lines file

File format for the ASCII fault center lines file:

```
n                ! Number of faults to be simulated from fault center lines
missing          ! The definition of the missing value
Fault name 1 of n
n_1              ! Number of fault center lines for this fault
x_11 y_11 z_11   ! Point 1 of n_1 points
dipD_11          ! Dip direction, 1 for east, 0 for west
dipA_11          ! Dip angle, 0 to 90 degrees
throw_11         ! Local throw
x_12 y_12 z_12   ! Point 2 of n_1 points
dipD_12
dipA_12
throw_12
  :
Fault name 2 of n
n_2
x_21 y_21 z_21   ! Point 1 of n_2 points
dipD_21
dipA_21
throw_21
  :
```

A.6 Well data files

We have collected below the different file formats currently used in HAVANA for input and output of well data.

A.6.1 Input of well paths

This file describes a single well path. It contains with N points specified with their (x, y, z) coordinates. The header consists of the integer number N. This number is ignored by Havana. Then follows N lines, each specifying the (x,y,z) coordinates of a point on the well path. The z coordinate is positive, indicating depth below the reference height. By default the reference height is the sea level, but if a non-zero height is given for the height of the kelly bushing, then this height is used. The well path starts directly above the first given point (at the reference height), ends at the last given point, and is linear between any pair of consecutive points.

Example:

```
2
460000 6580000 3000
461000 6581000 3200
```

See Section 1.15.12 for usage of such files.

A.6.2 Input of well observations of faults

This is an ASCII file that contains well observations of faults. Each line in the file represents one fault observation. The location of the observation may be given in two ways: Either, one may give the well name and the distance from Kelly Bushing, (i.e., the measured depth), or one may give the x, y, and z coordinates of the location. The program determines which option is used by determining whether the first item on the line is a text string or a number.

Following the specification of the location, there may be any number of items; as many as seven are read by the program. Each of these seven items must be either a number, or the character '?', which, of course, signifies missing data. If there are less than seven items on the line, the effect is the same as if the missing items had been '?'.

The seven items have the following meaning:

- Minimum fault throw at the observation point.
- Maximum fault throw at the observation point.
- Minimum dip azimuth of the fault (in degrees).
- Maximum dip azimuth of the fault (in degrees).
- Minimum dip of the fault (in degrees).
- Maximum dip of the fault (in degrees).
- Whether the fault is normal (signified by 1) or reverse (signified by 0).

To define the dip azimuth of a fault, take that normal vector to the fault plane that points upwards and project it to the horizontal plane. Then measure its angle (in degrees) with the vector pointing north (in the y coordinate direction), measuring the angle clockwise from north. This produces an angle between 0 and 360 degrees.

Examples:

```
460000 6580000 3000    8     8.2    130    132
461000 6581000 3200    ?     ?      124    125    88     ?     1
```

or, using measured depth to specify position:

```
A1     3945            8     8.2    130    132
A1     4164            ?     ?      124    125    88     ?     1
```

Note how the program reads the data line by line. Thus when there is no information for the last items on a line, it is not necessary to fill out the end of the line with question marks. See Section 1.15.13 for usage of these files.

A.6.3 Input of well observations of faults with depth uncertainty

This is an ASCII file that contains well observations of faults, just as in A.6.2 but with uncertainty in depth. The location of the observation must be given as the well name and the minimum and maximum distance from Kelly Bushing, (i.e., the measured depth).

Example:

```
A1     3945    3957    8    8.2    130    132
A1     4164    4187    ?    ?      124    125    88     ?     1
```

A.6.4 Input of well intersection thresholds

This file is used to specify intervals along well paths where there are no faults, or at least no faults with displacement above a certain threshold. Each line in the file corresponds to one such interval. The interval is specified by writing first the well name, and then the distances from Kelly Bushing (i.e. the measured depths) of the start and the end of the interval. The line is ended with a single number: The maximal displacement any fault intersecting the line can have (at the intersection point).

Example:

```
A1   0     3500  5
A2   0     2000  10
A2   2000  3500  5
```

A.6.5 Input of horizon wellpicks

This file is used for filtering of fault line control points. It contains coordinates of well picks from horizons. The first two columns are discarded when the file is read.

Example:

```
 TopC          Well_A          463174.625          5933349.000          1598.8566
 TopB          Well_E          460241.906          5935144.000          1625.7587
```

A.7 Havana-specific file type: Havana faults

The HAVANA faults directory is used for generating input to HAVANA 5.

HAVANA faults are stored in a directory containing the four files “version”, “Condition”, “EllFaults”, and “EllFaultGhosts”, and the four directories “IRAPfaults”, “IRAPghostFaults”, “ParametricFaults”, and “ParametricGhostFaults”. Elliptic, HAVANA-generated faults are stored in the “EllFaults” file,

The truncation rules are given in the file .truncTable. First is the number of faults to be truncated. Then, on each line, is the name of the fault to be truncated, followed by the number of faults and the fault names.

The other directories and files are always empty when generated with HAVANA 6.

Note that the names of the files and directories cannot be changed. When reading faults from a directory in the Havana faults format, the program will look for files and directories with the names described above, and ignore all other files and directories.

A.7.1 Format for the “EllFaults” file

The first number in the file is the number of faults the file contains. Then follows for each fault:

- The name of the fault.
- The position of the fault, in UTM/TVD coordinates.
- The length of the length, width and reverse drag axes of the ellipsoide. (Note that these numbers are half of the corresponding diameters of the ellipsoid; thus the length of the fault is twice the given number).
- The total maximal displacement of the fault.
- The asymmetry number (between -1 and 1) indicating how much of the displacement takes place on the hanging-wall side of the fault.

- A unit vector normal to the fault plane.
- A unit vector along the length of the fault. this vector will always have a zero z-component.
- A unit vector along the height of the fault. The displacement takes place along this vector.
- The family number of the fault.
- The number of planes truncating the fault. Then, for each such truncating plane, the indices of the faults truncating it.

A.8 Horizons

Horizons may be read and written on several different formats:

- STORM STORMGRID_BINARY format.
- RMS "CLASSIC" ASCII format.

The program will automatically recognize these formats.

A.9 Fault Blocks

Fault block information can be written out from RMS for every point in a horizon. RMS uses the following format:

```
Discrete  FaultBlock
465350.406  5930052.000  1545.000    8
465400.406  5930052.000  1545.000    8
465450.406  5930052.000  1545.000    8
465500.406  5930052.000  1545.000    8
```

The first three columns are x- y- and z- coordinates of the points, the last column is the fault block number.

The fault blocks must be defined in a separate file. This must be written manually. First column is fault block number, then all faults with side of fault for that block must be given. Only the faults that are defining a given fault block must be written in this file, with either hangwall (hw) or footwall (fw) side. Other faults in the fault set automatically gets "undef" side of this fault block.

```
1  F1 hw
3  F1 fw    F2 hw
8  F5 hw
9  F1 fw    F4 hw    F5 fw
10 F1 fw    F2 fw    F3 hw    F4 fw    F6 hw
12 F6 fw    F7 hw
13 F7 fw
14 F1 fw    F2 fw    F3 fw
```

A.10 Nodefile

The points in a nodefile can be read into Havana and moved according to commands given in ModifySurfaceFault.

```
-- This file defines a point set to be updated by moving faults.
 -- The points are top and base nodes from a corner point grid for pillar lines associated with faults.
 -- One pair of top and base point is specified per line with fault name, coordinates and a shift of coordinates.
 -- The file format is used as input/output to/from Havana.
 -- The modification of fault surfaces in Havana is used to calculate the shift of the nodes specified in this file.
 -- The file format is:
 -- Column 1:  The fault name.
 --            The fault name is the name specified in the input FAULTS keyword file for the
 --            Eclipse grid that is input to the grid updating program.
 --            A requirement is therefore that the Havana project use the same fault names
 --            as used in the Eclipse grid.
 -- Column 2:  Node obs number (integer).
 -- Column 3,4: I,J node index for the pillar line this corner point belongs to
 -- Column 5,6,7:  x,y,z coordinates of the top point (node) in global coordinates.
 -- Column 8,9:  deltaX, deltaY (the shift of the x and y coordinates for the top point after the fault is moved).
 --              These two numbers should be ignored by Havana as input, but is calculated by Havana
 --              and written to the output file.
 -- Column 10,11,12; x,y,z coordinates of the base point (node) in global coordinates.
 -- Column 13,14: delta X,deltaY (the shift of the x and y coordinates for the base point after the fault is moved).
 --              These two numbers should be ignored by Havana as input, but is calculated by Havana
 --              and written to the output file.
F1   0   11 8   168631.687500 569417.437500 1754.843994   0.0 0.0   168659.203125 569398.875000 1836.334961   0.0 0.0
F1   1   12 8   168660.531250 569417.750000 1754.982056   0.0 0.0   168687.843750 569400.125000 1836.313965   0.0 0.0
F1   2   12 9   168661.265625 569446.937500 1755.145996   0.0 0.0   168688.312500 569430.000000 1835.536011   0.0 0.0
```

A.11 RMS Internal Point Format

We use the format from RMS2010.

```
String    FaultTag
Float     VerticalSep
461674.174     5929715.370    2122.258       "fw F1" UNDEF
461674.156     5929706.245    2137.951       "fw F1" UNDEF
461657.138     5929715.049    2107.089       "hw F1" UNDEF
461670.600     5929715.101    2119.513       "hw F1" UNDEF
461703.690     5929715.205    2149.951       "hw F1" UNDEF
461792.751     5929715.363    2232.304       "hw F1" UNDEF
461851.181     5929748.293    2230.288       "hw F1" UNDEF
462134.406     5929907.184    2207.154       "hw F1" UNDEF
```

A.12 Internal Havana Format

This format is used to store a fault set of SFM faults and to be communicated between different modules in Havana. A separate directory is used with a number of files. One file is named `FaultSet.txt` and the others are one file for each fault. The format of the file for each fault is the same as used for the export of the structural model from RMS in the RMS2011, Cohiba version.

The `FaultSet.txt`-file has the following contents:

- Missing value
- Bounding box: xMin, yMin, zMin, lX, lY, lZ, 0
- Dimensions of truncation matrix
- Truncation matrix
- Number of faults
- For each fault: First line: Indicator saying if the fault has been changed. Second line: Totally 11 items:
    1. The file name of the file containing the data for the fault
    2. The name of the fault
    3. Type of fault (normal/reverse)

4. Missing value for fault
5. HW/FW distribution of the displacement
6. Displacement range
7. Grid x-min value
8. Grid y-min value
9. Grid length in x-direction
10. Grid length in y-direction
11. Indicator telling if the fault has displacement data
12. First parameter for relationship between maximum displacement and length of fault.
13. Second parameter for relationship between maximum displacement and length of fault.
14. Parameter for relationship between length and height of fault.

```
# Bounding box
459206.482 5929816.17 1463.52214 7762.4043 7672.13818 866.3927 0
# Truncation rules
7 7
0 0 0 0 0 0 0
1 0 0 0 0 0 0
1 1 0 0 0 0 0
0 0 2 0 0 0 0
0 0 0 0 0 0 0
0 0 2 0 0 0 0
0 0 0 0 0 0 0

# Number of faults
7
# FORMAT for each fault:
# Is changed (1 means fault is changed, 0 means unchanged)
# Filename - Fault name - Slip type - Distribution - Range - Xmin - Ymin - lx - ly - Has displacem
0
F1.grid F1 0 0.7 1686.09419 -4614.79831 -3004.21555 10600.0002 7500.00072 1 1 0.01 2
0
F2.grid F2 0 0.5 1000 -6749.79832 -648.790323 11000 1100 1 1 0.01 2
0
F3.grid F3 0 1 500 -3698.71561 -669.357798 8700.00008 1100 1 1 0.01 2
0
F4.grid F4 0 0.5 1000 -4897.01488 -719.402949 7399.99993 999.99995 1 1 0.01 2
0
F5.grid F5 0 0.5 1000 -2363.69866 -702.054795 5000.00016 1100 1 1 0.01 2
0
F6.grid F6 0 0.5 1000 -2129.09092 -724 4600.00003 1100 1 1 0.01 2
0
F7.grid F7 0 0.5 1000 -1395.13515 -661.081081 2800.00009 1000 1 1 0.01 2
```

The file specifying the data for the fault contains the following: (Lines starting with # are comments)

- Transformation matrix specified by keyword: TransformMatrix4x4. The first line contains four numbers with the x-components of the strike vector, dip vector, normal vector and reference point. The second line contains the y-components, the third line the z-components and the fourth line the numbers 0 0 0 1. This matrix is concluded with the keyword EndTransformMatrix4x4.
- Number of cells. Two lines, with the first specified by NU and the number of cells in x-direction, and the second with NV and the number of cells in y-direction.
- Data grid. Starts and ends with keywords Data and EndData. In between, all the grid values for the fault surface, one value on each line.
- Displacement field. Starts and ends with keywords AttributeDisplacement and EndAttributeDisplacement. In between, all the grid values for the displacement, one value on each line.
- For the directory containing the original fault set, two more attributes are given, namely the fw-moved grid and the hw-moved grid. These are given on the same format as the other grids, but with keywords AttributeFW-moved/EndAttributeFW-moved and AttributeHW-moved/EndAttributeHW-moved respectively.

B License manager

Starting with version 5.1 Havana has a new license manager controlling the permitted users, the expiration date, and the available modules. The license manager is part of the havana program itself and hence it does not need any daemon running in the background. All you need is a license file.

Hence to run HAVANA you need

1. A license file, obtainable from `support-havana@nr.no`.
2. A model file
3. Either give the command

   ```
   unix> havana -l /full_path/license.file model.file
   ```

   or if the Havana installation script in Appendix C has been used to create a start-up script, or if the environment variable `HAVANA_LICENSE_FILE` is set to the full path of the license file, just type `unix> havana  model.file`

   where `unix>` is the unix shell prompt. If required, contact the local system manager to get instructions for setting the environment variable.

C Installation script and start-up script

Starting with version 5.2 Havana is delivered with a perl installation script called `install_havana`. In order to run this script, make sure that your perl installation is v5.8.0 or newer.

The installation script places the havana binaries and the license file in directories chosen by the user, creates a start-up script, and places a soft link in, e.g., `\usr\bin\havana`. The start-up script automatically keeps track of the license file location and selects the correct binary for the platform used.

With the installation script correctly set up users do not need to set the variable `HAVANA_LICENSE_FILE` any more.