DemystiFicatioN (v07) Manual

# Introduction

DemystiFicatioN is a software routine written in Fortran that allows to do multi-cycle quasi-dynamic, 2D, earthquake simulation with fluid injection in a discrete fault network. It is using rate and state friction. It is using space-time boundary element method coupled a finite volume method that allows for non-linear diffusion along the faults.

# Important notice

1. Due to development constrains, I cannot ensure the backward compatibility with future version of the software.
2. The installation can be cumbersome, because of the need of HDF5.

# Quick start

1. **Compilation**
2. Install gfortran
3. Install HDF5
4. In the directory “DemystiFicatioN/”, type: “make software”
5. **Make a simulation**
6. In the directory “wrapper”, open with python “BP6QD\_A.py”, “fault\_network.py” or “LSBB\_noloading.py” to create a simulation.
7. Modify the parameters as you want.
8. If everything went fine, it should create a new directory in “problems” where all the parameters are written as a HDF5 file “config”.
9. **Launch a simulation**
10. To launch a simulation just type in a terminal:

“./build/DemystiFicatioN problems/*name\_of\_your\_simulation*/”

*Notes:* please do not forget the “/” at the end.

1. **Read the simulation**
2. In the directory “read\_data\_matlab”, open “load\_data.m”. Then use the function “loadandprocessdata\_v03” to read the data. You need to specify manually the number of points used in the simulation.

# Variables (International Units)

**Hyperparameter:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| max\_it | The maximum number of time step | Integer(1) |
| stride\_time | Relative error tolerance for Newton Raphson | Real(1) |
| iprec | Accuracy for the Fast Multipole Method | Real(1) |
| Simuation\_name | Name of the simulation | text |
| Fracture\_mode | “modeII” or “modeIII” | text |
| Static\_kernel | Calculation of the kernels: “direct”, “fmm” (only for modeIII), or “hmatrix” | text |
| Tol\_solver | Tolerance for the time solver | Real(1) |
| Freq\_writing\_file | Frequence of writing the output file | Integer(1) |
| Permeability\_coupling | 0 or 1; not yet working | Integer(1) |

**Material:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| mu | Shear modulus | Real(1) |
| cp | P-wave velocity | Real(1) |
| cs | S-wave velocity | Real(1) |

**Loading**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| normal\_loading\_dot | Normal rate loading on the fault (Pa/s) | Real(1) |
| shear\_loading\_dot | Shear rate loading on the fault (Pa/s) | Real(1) |

**Diffusion:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| Rock\_comp | Compressibility of the rock matrix | Real(1) |
| Fluid\_comp | Compressibility of the fluid | Real(1) |
| Fluid\_density | Density of the fluid | Real(1) |
| porosity | Porosity of the rock | Real(1) |
| Dyn\_viscosity | Dynamic viscosity of water | Real(1) |

**Rate-and-state:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| a | parameter a of rate and state | Real(N) |
| b | parameter b of rate and state | Real(N) |
| Dc | Critical slip distance | Real(N) |
| f0 | Friction coefficient at reference velocity V0 | Real(N) |
| V0 | Reference velocity | Real(N) |

**Fault geometry:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| Nb\_element | Normal of element | Integer(1) |
| node | Node(1, nb\_element+1): x-coordinate of the fault geometry  Node(2, nb\_element+1): y-coordinate of the fault geometry | Real(2,nb\_element+1) |

**Initial parameters:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| sigmaN | Initial normal traction (compression negative) | Real(N) |
| theta | Initial theta parameter on the fault | Real(N) |
| V | Initial velocity on the fault | Real(N) |
| P | Initial pressure | Real(N) |

**Injection:**

|  |  |  |
| --- | --- | --- |
| Variable name | Meaning | Size (2D) |
| nb\_source | Number of injection points | integer(1) |
| Q | Flow rate of injection (m/s) | Real(1) |
| t\_injection\_beg | Beginning time of the injection | Real(1) |
| t\_injection\_end | End time of the injection | Real(1) |
| index\_injection | Index of the injection point on the fault (Fortran way of indexing) |  |

# Advanced users: structure of the code

The code is organized as following:

master.f90:

Very short file calling sequentially to

* Read configuration files (input\_output.f90)
* Initialise arrays (initialisation.f90)
* Solve the seismic cycle and write results (solver.f90)

input\_output.f90:

File that is reading all the input files and display some information on the terminal

initialisation.f90:

File that initialize and allocate all the arrays that are not in input file. It also pre-computes the kernels in the fully dynamic case.

solver.f90:

File that is solving the seismic cycle.

special\_functions.f90:

Contains the Struve functions and Bessel functions.

variables.f90:

Contains all the variables that are made global in this program.

# Reading data (MATLAB) in “wrapper”

For reading the hdf5, we will need to use: “loadandprocessdata(*directory\_path*,'V','P','theta')”

MATLAB function that is reading the HDF5 files in *directory\_path*

“load\_input(*directory\_path*)”

MATLAB function that is reading the input file in *directory\_path*

Example of application is given in “example.m”

# Creating input file

# Bug reports and improvements

Pierre ROMANET

Please send an email to [romanet@geoazur.unice.fr](mailto:romanet@geoazur.unice.fr)

I will try my best to answer all reported bug and query for improvement

# Citation

If you are using the code, you are kindly asked to cite:

# In this version

* Sparse representation of conjugate gradient
* Harmonic average of permeability does not depend on gridsize, which makes a requirement of same discretization length for all the fault