FastCycles Manual

# Introduction

FastCycles is a software routine written in Fortran that allows to do multi-cycle, fully-dynamic or quasi-dynamic, 2D or 3D earthquake simulation. It is using rate and state friction. It is using the spectral boundary element method (Guebelle and Rice, 1995; Lapusta et al., 2000) together with a small slope approximation (Romanet et al., 2020, Romanet and Ozawa 2022) that allows to account for slight non-planar fault geometry.

Current version: the fully-dynamic non-planar fault geometry is disactivated in this version, this is because it seems that there was a typo in Romanet & Ozawa 2022. The quasi-dynamic non-planar is working. It will be added once the typo has been corrected.

# 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 and ifort. In future development, gfortran will be tested.

# Quick start

1. **Compilation**
2. Install ifort
3. Install HDF5
4. In the directory “FastCycles/2D/” or the directory “FastCycles/3D/”, in a terminal, type: “make software”
5. **Make a simulation**
6. In the directory “wrapper”, open with matlab “create\_example”.
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 list in text files.

*Notes:* You can also write your own routine if the output is similar to the example provided in the directory “problems/test/”.

1. **Launch a simulation**
2. To launch a simulation just type in a terminal:

“./fastcycles 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

**Hyperparameter:**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| max\_it | The maximum number of time step | Integer(1) | Integer(1) |
| omp\_threads | The number of openmp threads | Integer(1) | Integer(1) |
| rho\_c | Frequency cutting parameter (Lapusta and Liu, 2009, eq. 8) | Real(1) | Real(1) |
| eta\_w | Frequency cutting parameter (Lapusta and Liu, 2009, eq. 4, equivalent to ) | Real(1) | Real(1) |
| beta\_min | Parameter that controls the minimum time step in the fully dynamic method (Lapusta and Liu, 2009, eq. B2, equivalent to ) | Real(1) | Real(1) |
| gamma | Not used anymore, will be deleted soon | Real(1) | Real(1) |
| nr\_accuracy | Relative error tolerance for Newton Raphson | Real(1) | Real(1) |
| tol\_solver | Absolute error tolerance for time solver | Real(1) | Real(1) |
| quasi\_dynamic | 0: fully dynamic simulation 1: quasi dynamic simulation | Integer(1) | Integer(1) |

**Material:**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| mu | Shear modulus | Real(1) | Real(1) |
| cp | P-wave velocity | Real(1) | Real(1) |
| cs | S-wave velocity | Real(1) | Real(1) |

**Rate-and-state:**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| a | parameter a of rate and state | Real(N) | Real(N1,N2) |
| b | parameter b of rate and state | Real(N) | Real(N1,N2) |
| Dc | Critical slip distance | Real(N) | Real(N1,N2) |
| f0 | Friction coefficient at reference velocity V0 | Real(N) | Real(N1,N2) |
| V0 | Reference velocity | Real(N) | Real(N1,N2) |

**Fault geometry:**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| height | Height of the fault (it has to be sufficienly continuous to ensure that the second order derivative is defined) | Real(N) | Real(N1,N2) |
| mask | 0: the velocity is set to V\_mask  1: the fault follows rate and state | Integer(N) | Integer(N1,N2) |

*Note:* mask may look like 0 0 0 0 1 1 1 0 0 0 0; the 1 are where the fault is allowed to move following rate and state. V\_mask can be set to 0, then the fault is locked on the side where mask=0. V\_mask can be set to plate velocity, then the fault is loaded with plate velocity, where mask=0.

**Loading of the fault**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| V\_mask | Velocity set in the area where mask=0 | Real(1) | Real(1) |
| normal\_loading\_dot | Normal traction loading along the fault | Real(N) | Real(N1,N2) |
| shear\_loading\_dot | Shear traction loading | Real(N) | Real(N1,N2) |

**Initial parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Meaning | Size (2D) | Size (3D) |
| sigmaN | Initial normal traction | Real(N) | Real(N1,N2) |
| theta | Initial theta parameter on the fault | Real(N) | Real(N1,N2) |
| V | Initial velocity on the fault | Real(N) | Real(N1,N2) |

# 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)

# Creating input file

# Bug reports and improvements

Pierre ROMANET

Please send an email to [romanet@bosai.go.jp](mailto:romanet@bosai.go.jp)

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:

Romanet, Pierre, Dye SK Sato, and Ryosuke Ando. "Curvature, a mechanical link between the geometrical complexities of a fault: application to bends, kinks and rough faults." *Geophysical Journal International* 223.1 (2020): 211-232. DOI: 10.1093/gji/ggaa308

Romanet, Pierre, and So Ozawa. "Fully Dynamic Earthquake Cycle Simulations on a Nonplanar Fault Using the Spectral Boundary Integral Element Method (sBIEM)." *Bulletin of the Seismological Society of America* 112.1 (2022): 78-97. DOI: 10.1785/0120210178