HBI User's guide

So Ozawa

December 29, 2023

1 Introduction

Fortran90 software `HBI` performs three-dimensional quasi-dynamic numerical simulations of sequences of earthquakes and aseismic slip (SEAS). It can handle arbitrary fault geometry such as branches, step-overs, and rough faults in an elastic half-space. 2D problems can also be solved in the same code. The computation is accelerated by taking advantage of H-matrices. HACApK, open-source software for H-matrices, is called for constructing and operating H-matrices.

The latest versions can be found at `https://github.com/sozawa94/hbi`. Feel free to contact So Ozawa (`sozawa@stanford.edu`), if you have any questions.

2 License

This software is freely available under the MIT license. If you write a paper using this code, please cite the following manuscript.

So Ozawa, Akihiro Ida, Tetsuya Hoshino, Ryosuke Ando (2023), "Large-scale earthquake sequence simulations of 3D geometrically complex faults using the boundary element method accelerated by lattice H-matrices on distributed memory computer systems", Geophysical Journal International

3 How to run

Download the source codes from github. Type:

```
git clone http://github.com/sozawa94/hbi
```

To compile and link the code, type

```
make
```

You may need to modify `Makefile` depending on your environment.

To run a simulation, type:

```
./lhbiem (inputfile)
```

or use MPI parallelization;

```
mpirun -np 16 ./lhbiem (inputfile)
```

Some examples of input files are included in the directory. To run the code in parallel, **the number of MPI processes must be a squared number.**

As an example, `examples/2dp.in` and the corresponding geometry and parameter files are included. Several input files for SEAS project (`https://strike.scec.org/cvws/seas/`) are also available.

The standard output first shows the information about (lattice) H-matrices (approximated integration kernel). Then, timestep and time (measured by year) are recorded. Other output files are mentioned later.

4 Problem setting

4.1 Description of Input file

Different simulations can be performed by modifying the input file, rather than modifying the source code. The input variable list is:

name	type	meaning
jobnumber	integer	job number (name of output files)
problem	character	dimension, planar or non-planar, mesh, etc.
ncellg	integer	number of elements (not used in 3dp, 3dph, 3dht,3dnt)
nstep	integer	maximum number of time steps
interval	integer	output & checkpointing every this time steps
dtout	real	output every this time (year)
dtinit	real	initial time step width
geometry_file	character	name of geometry file
parameter_file	character	name of parameter file
parameterfromfile	logical	true if parameters are read from parameter_file
restart	logical	true if restarting a suspended job
backslip	logical	true if loading is added by backflip
vpl	real	the velocity of backslip (m/s) (used if backflip=T)
sr	real	the stressing rate (MPa/s) (used if backslip=F)
tmax	real	maximum time (s)
sigmaconst	logical	true if normal stress is kept constant
ds	real	element size (km) (used in 2dp, 3dp, 3dph)
imax	integer	the number of elements in slip-parallel direction (used in 3dp, 3dph)
jmax	integer	the number of elements in slip-perpendicular direction (used in 3dp, 3dph)
crake	real	rake angle (deg) in 3D problems (not used if parameterfromfile=T)
dipangle	real	dip angle (deg) used in 3dph and 2dph
evlaw	character	evolution law for RSF (”aging” or ”slip’, default=”aging”)
a	real	a in RSF (not used if parameterfromfile=T)
b	real	b in RSF (not used if parameterfromfile=T)
dc	real	dc in RSF (not used if parameterfromfile=T)
f0	real	f0 in RSF (not used if parameterfromfile=T)
tauinit	real	initial shear stress (MPa) (not used if parameterfromfile=T)
sigmainit	real	initial normal stress (MPa) (not used if parameterfromfile=T)
velinit	real	initial slip rate (m/s) (not used if parameterfromfile=T)
eps_h	real	error allowance in H matrix (default: 1e-4)
eps_r	real	error allowance in Runge-Kutta (default: 1e-4)
velmax	real	computation stop if maximum velocity is above this value (default:1e7)
velmin	real	computation stop if maximum velocity is below this value (default:1e-16)
limitsigma	logical	true if you want to limit normal stress
maxsig	real	maximum normal stress (MPa) (used if limitsigma=T)
minsig	real	minimum normal stress (MPa)(used if limitsigma=T)

4.2 Problem

The list of problems are follows:

problem	description
2dp	2D planar fault in full-space (plane strain)
2dph	2D planar fault in half-space (plane strain)
2dn	2D nonplanar fault in full-space (plane strain)
2dnh	2D nonplanar fault in half-space (plane strain)
25d	2D nonplanar fault in full-space (plane strain) with finite fault width (2.5D approximation)
3dp	3D planar squared elements in full-space
3dph	3D planar squared elements in half-space
3dnt	3D nonplanar unstructured triangular elements in full-space
3dht	3D nonplanar unstructured triangular elements in half-space

4.3 Physical variables and constants

The following units are assumed:

quantity	unit
time	s
location	km
displacement	m
slip velocity	m/s
wave speed	km/s
shear modulus	GPa
stress	MPa

The fundamental physical constants are set in `m_const.f90`. The default values are

quantity	value
rigidity	32.04GPa
P-wave speed	6.000 km/s
S-wave speed	3.464 km/s
reference slip rate on RSF	1μm/s

4.4 Fault Geometry

For 2dp, 2dph, 3dp, and 3dph, fault geometry is set in inputfile. For other problems, an additional `geometry_file` is necessary.

4.4.1 2dp

The fault geometry is uniquely characterized by its element size `ds` and the number of elements `ncellg`.

4.4.2 3dp

The fault geometry is uniquely characterized by its element size `ds` and the number of elements along strike and dip `(imax, jmax)`.

4.4.3 2dph

The fault geometry is characterized by its element size `ds`, the number of elements `ncellg`, and dip angle in degree `dipangle`. Currently, only planar geometry is supported.

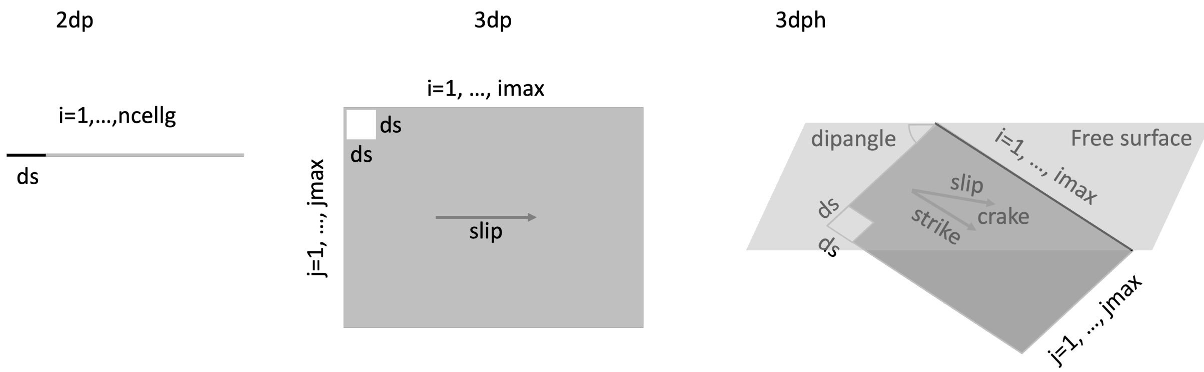

Figure 1: Fault geometry

4.4.4 3dph

The fault geometry is characterized by its element size `ds`, the number of elements along strike and dip `(imax, jmax)`, and dip angle in degree `dipangle`. See also Figure 1.

4.4.5 2dn, 2dnh, 25d

The coordinates of two edges of a straight line have the geometrical information of an element. `xel<xer` is necessary. `geomerty_file` has the following format:

```
xel(1) xer(1) yel(1) yer(1)
xel(2) xer(2) yel(2) yer(2)
...
```

For 2dnh, the structure `geomerty_file` is the same with 2dn. Here, $y = 0$ is assumed to be the free surface and y-axis is downward (y is always positive).

4.4.6 3dnt, 3dht

The coordinates of the three edges of a triangle have the geometric information of an element. `geometry_file` should have STL (ASCII) format:

```
facet normal normal_vector(1:3)
  outer loop
    vertex xs1 ys1 zs1
    vertex xs2 ys2 zs2
    vertex xs3 ys3 zs3
  endloop
endfacet
```

This structure repeats `ncellg` times. For `3dht`, $z = 0$ is the free surface and all elements must be located under the free surface.

4.5 Parameters and Initial conditions

The following parameters and initial values are necessary for each element.

4.5.1 Rake angle

In 3D problems, it is also necessary to specify the direction of slip (rake angle) for each element. The angle of rake is defined as 90 = reverse faulting, 0 = right lateral strike-slip faulting, and -90=normal faulting. The code does not allow for a time-varying slip direction. In the uniform case, set `crake` in `inputfile`. 2D problems do not use rake values.

4.5.2 Friction parameters

`HBI` uses the regularized version of the rate and state friction law (RSF), which is employed in the SEAS benchmark problems (Erickson et al., 2020). The friction coefficient are given by

$$\tau/\sigma = a\sinh^{-1}(\mathrm{V}e^{\psi/\mathrm{a}}/2\mathrm{V}_0). \tag{1}$$

The state evolution is governed either by the aging law

$$\frac{d\psi}{dt} = \frac{bV_0}{d_c}\exp[(f_0 - \psi)/b] - \frac{bV}{d_c}. \tag{2}$$

or the slip law

$$\frac{d\psi}{dt} = -\frac{V_0}{d_c}(f - f_{ss}) \tag{3}$$

$$f_{ss} = f_0 + (a-b)log\frac{V}{V_0} \tag{4}$$

The values of parameters (a, b, d_c, f_0) are set in `inputfile` in the case of uniform parameters.

4.5.3 Initial conditions

The initial condition is uniquely determined by the value of normal stress, shear stress, and slip rate at each element (the state variable is determined by these three accordingly). In the case of uniform initial values of (σ, τ, V), these values are set in `inputfile` as `sigmainit`, `tauinit`, and `velinit`.

4.5.4 External loading

To have repeated ruptures on the same fault, external loading is necessary. `taudot` and `sigdot` are the stressing rates for shear and normal stresses, respectively. In the uniform case, `taudot` is `sr` and `sigdot` is zero. the value of `sr` must be specified in `inputfile`.

Alternatively, the code has "backslip" loading option. If `backslip=T`, then the loading is added by slip deficit rate, that is,

$$\dot{\tau}_i = -V_{pl}\sum_j K_{ij} \tag{5}$$

$$\dot{\sigma}_i = -V_{pl}\sum_j L_{ij} \tag{6}$$

where K_{ij} and L_{ij} are kernel matrices for shear and normal stresses, respectively. The value of `vpl` must be specified in `inputfile`. Spatially non-uniform backslip has not been implemented.

4.5.5 Format of parameterfile

If you want to use non-uniform parameters and initial values, set `parameterfromfile T` in `inputfile` and create `parameter_file`. `parameter_file` has following structure:

```
rake(1) a(1) b(1) dc(1) f0(1) tau(1) sigma(1) vel(1) taudot(1) sigdot(1)
rake(2) a(2) b(2) dc(2) f0(2) tau(2) sigma(2) vel(2) taudot(2) sigdot(2)
...
```

4.6 Stop control

The simulation stops when any of the following is satisfied.

1. the time-step reaches `nstep`
2. the maximum slip rate is greater than `velmax`.
3. the maximum slip rate is smaller than `velmin`.
4. the physical time reaches `tmax`.

4.7 Restart

If `restart=T`, then the code reads the distributed output file to set the initial condition and restarts the simulation from the last checkpoint (the last time field variables were saved). The description of `inputfile` must be the same (other than `restart`), including the job number and other parameters.

4.8 Useful functions

If `sigmaconst=T`, then normal stresses do not change over time as assumed in many studies. If `sigmaconst=F` (as the default value), the normal stress changes with time. The computation can fail if the normal stress becomes very large or small. To avoid this, the user can set limits on the value of normal stress by setting `limitsigma=T`. In this case, `maxsig` and `minsig` are the maximum and minimum normal stresses, which should be stated in `inputfile`.

5 Visualization

For those familiar with Python, see Analysis2D.ipynb and Analysis3D.ipynb.

ASCII files `monitorX.dat` and `eventX.dat` have global information.

`monitorX.dat` records the following information at every time step.

```
timestep time max(log10(vel)) mean(disp) mean(tau/sigma) max(sigma) min(sigma)
error-of-RK stepwidth wall-clock-time
```

For example, if you want to view the temporal change of the maximum slip rate in `gnuplot`, type

```
pl 'monitorX.dat' u 2:3 w l
```

The earthquake catalog is saved in `eventX.dat`. This file lists the event number, the onset time, and the moment magnitude. In 2D problems, the moment magnitude does not have meaning. //

The field data are saved in binary stream files. `xyzX_R.dat` is the coordinates of the elements. `velX_R.dat`, `tauX_R.dat`, `sigmaX_R.dat`, and `slipX_R.dat` are slip velocity, shear stress, normal stress, and slip every `interval` time steps, respectively. These are binary stream files.

5.1 Scripts gathering distributed outputs in parallel simulations

The code outd.f90 gathers distributed output files into a single unified ASCII file. Use it like

```
ifort outd.f90 -o outd
./outd X N
```

where `X` is the `jobnumber` and `N` is the number of output files (0 - N-1), where $N = \sqrt{N_p}$ and N_p is the number of MPI processes. `fieldX.dat` will be generated.

`fieldX.dat` has the following structure.

```
x(1) y(1) z(1) vel(1) tau(1) sigma(1) disp(1)
x(2) y(2) z(2) vel(2) tau(2) sigma(2) disp(2)
...
x(ncellg) y(ncellg) z(ncellg) vel(ncellg) tau(ncellg) sigma(ncellg) disp(ncellg)
---blank line---
---blank line---
```

```
x(1) y(1) z(1) vel(1) tau(1) sigma(1) disp(1)
x(2) y(2) z(2) vel(2) tau(2) sigma(2) disp(2)
...
x(ncellg) y(ncellg) z(ncellg) vel(ncellg) tau(ncellg) sigma(ncellg) disp(ncellg)
---blank line---
---blank line---
...
```

You can visualize the result with gnuplot using this file like:

```
pl 'fieldX.dat' u 1:3:4 index 5 lc palette pt 7
```

This will show the snapshots of the slip rate at time step=5*`interval`.

5.2 Paraview

6 References

Tse & Rice (1986) and Rice (1993) are early examples of earthquake cycle simulations on a planar fault with rate and state friction. For nonplanar fault geometry, Segall (2010) is a great reference to the theoretical foundation of the BEM kernel. For the implementation of `2dn`, we use the formula in Ando et al. (2007). 3dnt/3dht is based on the expression of Nikkhoo & Walter (2015). The time-stepping algorithm is based on "Numerical Recipe for Fortran90" (Press et al., 2007).

7 Acknowledgements

I learned a lot about the method from Nobuki Kame and Makiko Ohtani when I was an undergraduate student. The basic structure of `HBI` is derived from their codes. HACApK library used in `HBI` was primarily developed by Akihiro Ida. I also thank Ryosuke Ando, Brittany Erickson, Eric M. Dunham, Pierre Romanet, Tetsuya Hoshino, and Ryoya Matsushima for their suggestions that improved the performance, accuracy, stability, and readability of the code.