

PEPC User Guide

Paul Gibbon

John von Neumann Institute for Computing, ZAM,
Forschungszentrum Jülich GmbH, D-52425 Jülich, Germany

presented on the occasion of

December 13, 2005

Contents

1	Introduction	3
2	Compiling pepc	3
3	Running the code	4
4	Input parameters and switches	6
4.1	Plasma configuration: plasma_config	6
4.2	Geometries: target_geometry	6
4.3	Integrator schemes:scheme	6
4.4	Particle, laser beam or external fields: beam_config.in	6
4.5	Run control: I/O switches	7
5	Handling the output data	8
5.1	Particle data	9
5.2	Gridded data	9
5.3	Tree kernel diagnostics:	9
6	Using the GMT imaging software for 2D plots	10
7	Online visualization with VISIT and XNBODY	13

PEPC

PEPC | Pretty Efficient Parallel Coulomb-solver | is a parallel tree-code for rapid computation of long-range Coulomb forces, based on the original Barnes-Hut

- > make cleanlib
- > make lpepc
- > make cleanapp
- > make pepcb

The library should always be compiled first because header files are needed by pepcb, and this step ensures that these are copied over (or freshly linked) from the lpepcsrc directory.

At this point there may well be linker errors because of missing libraries, such as MPI or from

```

plasma_config = 1
target_geometry = 1
theta = 0.5
Te_keV = 0.5
Ti_keV = 0.
mass_ratio = 500.
q_factor = 1.
coulomb = .true.
lenjones 46.9619 0 Td (=)Tj 10.5502 0 Td (1.).false.
bond_const = 2.e-3
r_sphere 46.9619 0 Td (=)Tj 10.5502 0 Td (1.)4
ma = 1.
ma = 2.
ma = 2.

```

```

onfig_in=0
0.1
= 0.5
= 6.
= 20.
= 1.0

```

```

1
0
3

```

r e e s r s r e s

P r c e d

Particle data is output independently by each CPU to avoid memory and MPI

These and other .gle files can be viewed using the graphics program GLE (Graphics Layout Engine), currently available from:

<http://glx.sourceforge.net>

1. single M A source e to Dos

peratures, low velocity and temperature. Graphical output from PEPC can be created using the postprocessor slicer, which converts the particle data into 'fluid' quantities such as mass density

```

        ymi n= 200.0      box ymi n
        ymax= 700.0      box ymax
        yti ck= 100.0    interval
zmi n= -50.0      box zmi n
zmax= 150.0      box zmax
zti ck= 100.0    interval
```

```

xmi n= -5.0      box vxmi n for phase space plots
xmax= 5.0      box vmax
tick= 2.5      interval
ymi n= -5.0      box vymi n
```

xbox= 2.

Then:

```
> make_snaps di sc1 runpp plot
```

**will perform the postprocessing and produce plots at timesteps 100, 200 and 300. The
rst parameter is an arbitrary run label which will get stamp ed on the corner of the plots**