The PIC code BOPS

Description

- BOPS is a one-and-three-halves (1 spatial, 3 velocity coordinates: 1D3V) particle-in-cell code originally created by Paul Gibbon and Tony Bell in the Plasma Physics Group of Imperial College, London.
- Based on standard algorithm for a 1D electromagnetic PIC code from Birdsall & Langdon.
- Optionally employs a Lorentz 'boost' along the target surface to mimic 2D, periodic-in-y geometry, with big savings in compute time.
- Applications: absorption, electron heating, high harmonic generation, ion acceleration

Prerequisities

- Current version: BOPS 3.6
- Download site: `https://trac.version.fz-juelich.de/bops`
- Compiler: sequential code written in Fortran 90 (Intel ifort or GNU's gfortran)
- Run scripts provided are designed for generic Unix systems: Ubuntu, SuSE, RedHat, Fedora and also MacOS.

Linux and MAC users:
Unpack the tar file (name may differ) with:
`tar xvfz bops3.6.tar.gz` and cd to the installation directory `bops`.

Windows users

- First install a unix emulator such as CYGWIN (www.cygwin.com) or MinGW (http://www.mingw.org/). These offer a fully-fledged Unix environment emulator under Windows. In addition to the default tools/packages offered during the installation you should make sure you have:
- Gnu compilers gcc, g77, g95 etc. (*devel* package)
- make (*devel* package)
- vi, emacs (*editors* package)
- X11 libraries if you want to generate graphics directly under cygwin (eg using gnuplot)
- For Cygwin if you forget anything first time, just click on the Cygwin installer icon to locate/update extra packages. MinGW should have everything you need.
- Download and unpack the bops3.6.tar.gz file with an archiving tool (eg PowerArchiver: www.powerarchiver.com). Put this in your 'home' directory and unpack the tar file there.
- Open a Cygwin or MinGW terminal/shell and 'cd' to the bops directory.

Compiling and Running

Installation
The directory structure resulting from unpacking the tar files should look like this:

`src/`	...	fortran90 source code
`doc/`	...	documentation
`tutorial/`	...	scripts and files for tutorial
`benchmarks/`	...	additional sample scripts
`tools/`	...	postprocessing tools

To compile:
Go to the source directory `src`, and edit the `Makefile`. Adjust and tune the flags to match your machine type (FC=gfortran etc).
Then do:

```
make
```

Running BOPS
Once the code has compiled, go back up to the base (or top) directory and enter the `tutorial` directory. Here you will find the run scripts (suffix .sh) which launch the example simulations. These can be edited directly or copied as needed.
To run from this directory, just type eg:

```
./foil.sh
```

Or if you prefer to stay in the top directory (paths in foil.sh may need adjusting first):

```
cp tutorial/foil.sh .
./foil.sh
```

Further examples can be found in `benchmarks`

Graphics

A few scripts for GLE, Gnuplot and Matlab are provided in the tools directory to help view the data.

Automatic postprocessing with GLE
Assuming you have GLE installed (see http://glx.sourceforge.net), these plots can be postprocessed with the help of a program in the `tools` subdirectory.

```
cd tools/gle
make od2gle
```

Now go back up to the top directory and run the shell script

```
./odpp resabs1
```

where, for example, `resabs1` is the name of the run directory you wish to process. In case you execute this from a different 'base directory', make sure the odpp script can find the od2gle binary (eg include it in your PATH). The program will scan the file `foil.id` for plots – the numbers correspond to the IDs in the tables below – and attempt to produce series of line graphs for these in `resabs1/plots`. These can be viewed with either graphics or pdf viewer.

Organising your simulations
If you are performing a series of runs, e.g. with different laser parameters, you may want to archive the data under a particular topic. All of the above steps can be performed within a subdirectory (e.g.: bops/harmonics), but may need some adjustments to the execution and postprocessing scripts.

Project I: Laser wakefield accelerator

1. Edit and run the script: `./wake.sh`. Note that the input parameters are normalised to laser wavelength and wavenumber.
2. Look at the printed output to check the actual laser and plasma parameters (are they what you intended?)

Plot files

These are in run directory (`foil_tnsa1`) in ASCII format and have a suffix NN.xy, where NN is the snapshot number. See `bops.oddata` for complete list

`ezsi`	EM field E_z (S-polarized wave)
`nenc`	Electron density
`exsi`	Electrostatic field
`pxxe`	Electron momenta ($x - p_x$ phase space)

3. Examine the field and particle phase space plots at successive time intervals (e.g.: `t=100, t=200`). The gnuplot script `wake.gp` should help you get started.
4. How can the plasma wave amplitude $\mid E_x \mid_{\max}$ be optimised? Hint: try changing the electron density and/or pulse length.
5. Increase the laser amplitude to ($a_0 = 3$) and observe the difference in the electron phase space and plasma wave. How does the matching condition need to be altered in this regime? Why is the amplitude of the latter reduced towards the back of the wake?
6. (advanced) Set up a plasma equivalent to twice the dephasing length (Eq. **??**) and determine the maximum energy of electrons trapped in the plasma wave. Can you beat the scaling predicted by Eq. **??**?

Project II: Ion acceleration – TNSA vs. RPA

1 **TNSA regime**.
a) The script `foil.sh` sets up a 2 μm 'foil' out of frozen hydrogen ($Z = A = 1$), with density $n_e/n_c = 36$. This is irradiated by a linearly polarized 50 fs pulse with intensity 10^{19} Wcm^{-2} .
Run the script and inspect the following plots at the snapshot times (0, 50, 100, 150) fs:

Plot files

`nenc, ninc`	Electron & Ion densities
`exdc`	*Cycle-averaged* electrostatic field
`pxxe, pxxi`	Electron & ion momenta
`fuep, fuip`	Electron, ion energy spectra
`uinc, ubac`	Incoming and outgoing wave energy

b) Try varying, eg: pulse amplitude or duration and compare the scaling of the maximum ion energy against the theoretical prediction of Eq. (**??**).

2 **HB regime**.
a) Copy the script to a new file (eg: `hb.sh`), switch the polarization from linear to circular (set `cpolzn='C'`), change the run directory – eg: `RUN=hb1` and rerun. (What phase space variables can you check the laser polarization with?) Compare the results to the TNSA case.

b) To compare with HB theory, it is convient to specify a 'square' pulse profile. This can be done with, eg: `ilas=1, trise=5, tpulse=150`. Another important constraint in this case is to avoid numerical heating arising from an underresolved (cold) electron Debye length. Adjust the grid size/resolution to ensure that $\Delta x < 0.5\lambda_D$. Rerun and compare the ion shock velocity with Eq. (**??**).

3 **Light sail regime**.
a) Use the matching condition Eq. (**??**) to determine the foil width (using the same laser and target material) for which the light sail regime is reached. Set `ilas=4` with `tpulse=50, tfall=5`, and ensure that the new foil is resolved by the grid. Create a new script with these parameters and rerun. NB: you may have to increase the parameter `uimax` for this case!

b) Investigate the scaling behaviour of the monoenergetic ion feature with intensity, pulse length etc.

c) Now repeat with a Gaussian or sin^2 pulse shape. Compare the hot electron spectrum and phase space with the flat-top case at early times: what causes this difference? What could you change to improve the far-field ion sail stability for realistic pulse forms?

d) Now try the same thing with a carbon foil. Here, you will need higher (solid) number densities and multiply charged ions (see Eq. **??**). What happens to the max. ion energy? How does the energy/nucleon compare with the hydrogen case?