Computing Assignment for Aliya Babul
CTA2000, Summer 2016

May 3, 2016

Overview

In order to monitor the progress of numerical simulations of coalescing binary black holes (BBH), we use a Python program called MonitorBbhRuns.py. More specifically, it generates diagnostic plots which help monitor the following key attributes in BBH evolutions:

1. computational efficiency (speed & behavior of Adaptive-Mesh-Refinement (AMR)),
2. eccentricity, and efficiency of eccentricity reduction,
3. behavior near merger and ringdown,

The script takes a list of directories, each one is assumed to have the canonical sub-directory structure DIR/Ecc*/Ev*/Lev[0-9]_[A-Z][A-Z]/Run :

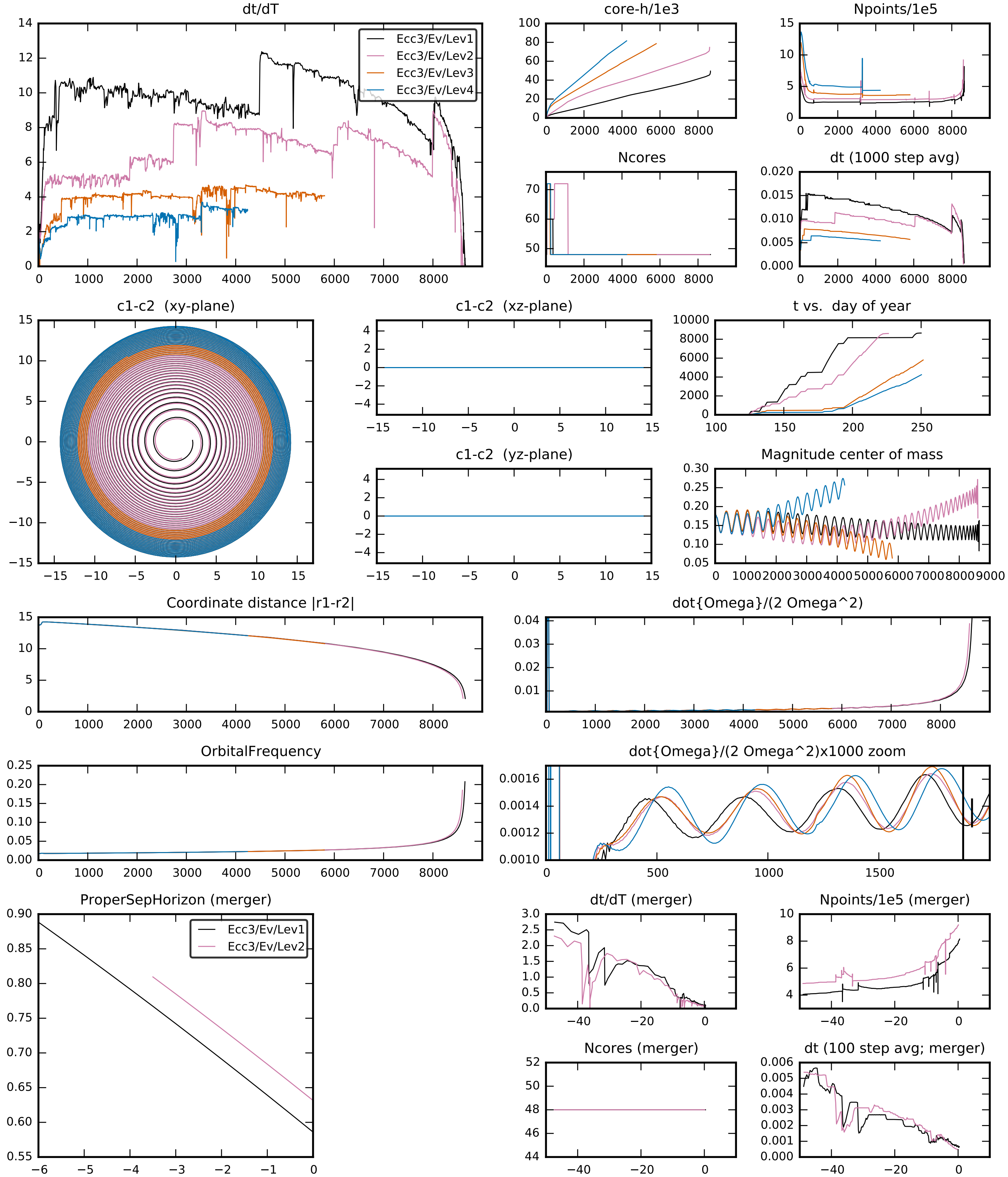
```
DIR
|- Ecc0
| |- Ev
| | |- Lev0_[A-Z][A-Z]
| | |- Lev0_Ringdown
| | |  Lev0_[A-Z][A-Z]
| | |- Lev1_[A-Z][A-Z]
| | |- Lev1_Ringdown
| | :    Lev1_[A-Z][A-Z]
| |
| |- Ev1
| |- Ev*
| :
|- Ecc1
|- Ecc*
:
```

where EccN contains data for the N^{th} iteration of binary's eccentricity reduction procedure, and within it Ev contains all information about the actual evolution in that iteration. Within Ev are evolution segments, named as LevN_MM. Here LevN indicates the numerical resolution of the run, with $N = 1 \rightarrow 5$ for increasing grid resolution. The trailing letters MM are for ordering the segments alphabetically.

For each configuration 'DIR', the script reads certain files from all Ecc*/Ev*/Lev* directories, and generates one pdf-page of plots. An example is shown below.

BBH_SKS_d13.78_q6_sA_0_0_0.960_sB_0_0_0

d=13.78 Omega0=0.017385 adot0=0.000160 q=6 chiA=(0, 0, 0.960) chiB=(0, 0, 0) (from Ecc3/Ev/EvID)



The plots fall into three categories:

1. **Top Quarter: Efficiency**
   Plots of dt/dT, core-hours, number of grid points, number of cores, time-step. These plots are meant to provide a quick overview of whether AMR refinement works ok, and a means to compare different configurations quickly, to compare efficiency of different BBH parameters, or different machines.

2. **Middle Two Quarters: Inspiral & Eccentricity Reduction**
   Plots of trajectories, coordinate distance between the centers of the AH, orbital frequency, and the adiabaticity parameter multiplied by 0.5: $E \equiv \dot{\Omega}/(2\Omega^2)$. The latter plot allows to assess eccentricity, because orbital eccentricity e results in center-to-peak oscillations of E that are comparable to e. Specifically to assess eccentricity reduction, a zoom-in plot of E is provided with y-axis scale adjusted to the *last Ecc** iteration. This zoom-in plot has two vertical bars at the right-hand-side, which indicate the peak-to-peak oscillations of E for eccentricity $1e-4$ and $1e-3$.

3. **Bottom Quarter: Merger & Ringdown**
   The bottom plots are only shown if at least one run of the configuration has made it to coordinate distance $|r_1 - r_2| < 6$. If that is the case, plots of proper separation, and efficiency are shown for the interval $t_{merger} - 49 <= t <= t_{Merger} + 199$ Here, t_{merger} is extracted for each run separately as the last data-point in `Trajectory_AhA.dat`. The end time (default: 199) can be adjusted with the option `--tmaxPostMerger`.

In addition, runs where the newest data is **younger than two days** are shown with **thick lines**. Older runs with thin lines. This aids the identification of crashed/stopped runs.

Assignment

Enhance `MonitorBbhRuns.py` through one *or* both of the following:

1. Provide early-inspiral estimates of time-to-merger. For ongoing runs, fit a post-Newtonian formula to orbital frequency. From the fit, predict what the merger time is, and predict how much longer a certain run is likely to take. This could already be done with about 1000M of inspiral, and would yield an early indication of how long runs take.

2. Provide near-merger estimates of time-to merger. Many runs crash within a few M of merger. The "ProperSepHorizon (merger)" plot is meant to give an indication of how close to merger each run is. However, this plot is created in an ad-hoc way currently. Instead, this panel could be constructed as follows. For each run where proper separation $s(t)$ gets sufficiently close to zero (say, $s(t) < 1$) do the following steps:

   (a) fit polynomial to $s(t)$ for small values of $s(t)$, say, the portion with $s(t) < 2$,

(b) find zero-crossing time, t_0, from the fitted polynomial, and

(c) using $t - t_0$ as the x-axis, plot $s(t)$.