POST-PROCESSING CACTUS SIMULATIONS WITH PYTHON

June 16, 2022

Gabriele Bozzola

Department of Astronomy and Steward Observatory,
University of Arizona

PLAN FOR THE DAY

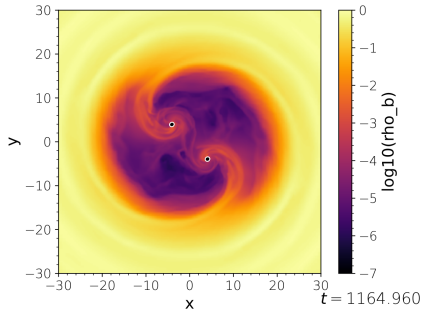
1. A brief general introduction on `kuibit`
2. Installing and setting up `kuibit`
3. Exploring examples
4. Questions and break
5. Developing a script for a new analysis

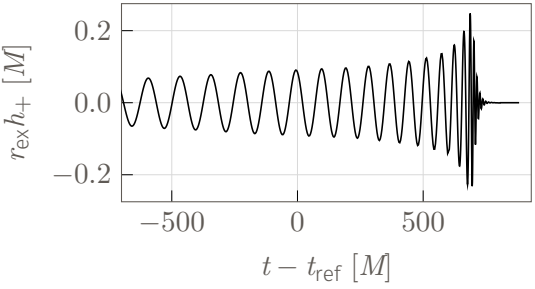
All the material is public, focus on understanding and ask questions

https://github.com/Sbozzolo/kuibit_ETUIdaho

KUIBIT IS A PYTHON LIBRARY FOR POST-PROCESSING SIMULATIONS

Post-processing = visualization & quantitative analysis





kuibit takes care of the low-level details and lets you focus on science

KUIBIT HAS EXCELLENT DOCUMENTATION (SBOZZOLO.GITHUB.IO/KUIBIT)

Getting started with SimDir

The `SimDir` class provide easy access to simulation data. Most data analysis start by using this object. `SimDir` takes as input the top level directory containing simulation data, and read and organizes the content. `SimDir` contains the index of all the information that is possible to extract from the ASCII and HDF5 files. If there are restarts, `SimDir` will handle them transparently.

Defining a SimDir object

Assuming `gw150914` is the folder where a simulation was run. `gw150914` can possibly contain multiple checkpoints and restarts.

```
import kuibit.simdir as sd

sim = sd.SimDir("gw150914")
```

USAGE

`kuibit.visualize_matplotlib.`**`plot_color`**(*data*, ***kwargs*) [source]

Plot the given data.

You can pass (everything is processed by `preprocess_plot_grid()` so that at the end we have a 2D NumPy array): - A 2D NumPy array, - A `UniformGridData`, - A `HierarchicalGridData`, - A `BaseOneGridFunction`.

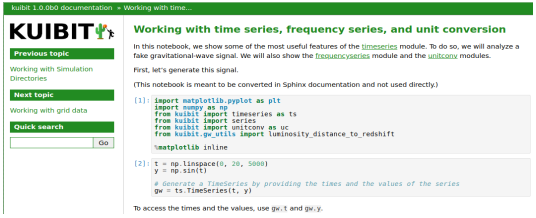
Depending on what you pass, you might need additional arguments.

If you pass a `BaseOneGridFunction`, you need also to pass `iteration`, and `shape`. If you pass `HierarchicalGridData`, you also need to pass `shape`. In all cases you can also pass `x0` and `x1` to define origin and corner of the grid. You can pass the option `resample=True` if you want to do bilinear resampling at the grid data level, otherwise, nearest neighbor resampling is done. When you pass the NumPy array, passing `coordinates` will argument will make sure that those coordinates are used.

All the unknown arguments are passed to `imshow`.

Parameters: • **data** (*2D NumPy array, or object that can be cast to 2D NumPy array.*) – Data that has to be plotted. The function expects a 2D NumPy array, but the decorator `preprocess_plot_grid()` allows it to take different kind of data.

APIs



TUTORIALS

Examples

Below you will find a list of examples to perform more or less common analysis. You can immediately start doing science without writing one line of code using these examples. The scripts provided can be used for plotting, extracting gravitational waves, or other useful information. To get the most out of these examples, check out the recommendations on how to use the examples page.

Note that all these examples contain a significant fraction of boilerplate that is needed to keep them general and immediately useful. When learning `kuibit`, you can ignore all of this.

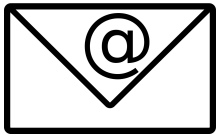
You can download these examples as archive from the GitHub release page, which is automatically updated with each release.

Scripts

- plot_1d_vars.py
- plot_ah_coordinate_velocity.py
- plot_ah_found.py
- plot_ah_radius.py
- plot_ah_separation.py
- plot_constraints.py

EXAMPLES

HOW TO GET HELP?

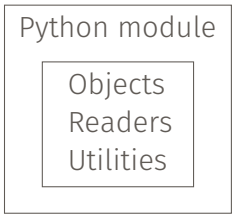


gabrielebozzola@email.arizona.edu
(fast)

t.me/kuibit
(fastest)

(instantaneous)

STRUCTURE OF THE `KUIBIT` PROJECT

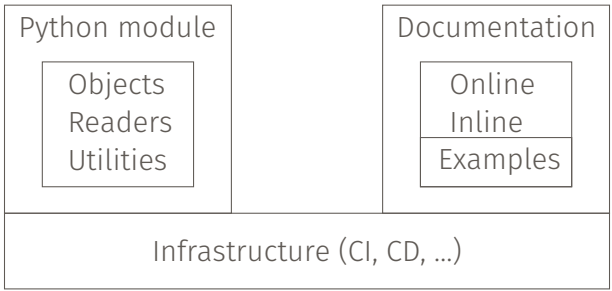


`kuibit` is workflow independent

STRUCTURE OF THE `KUIBIT` PROJECT

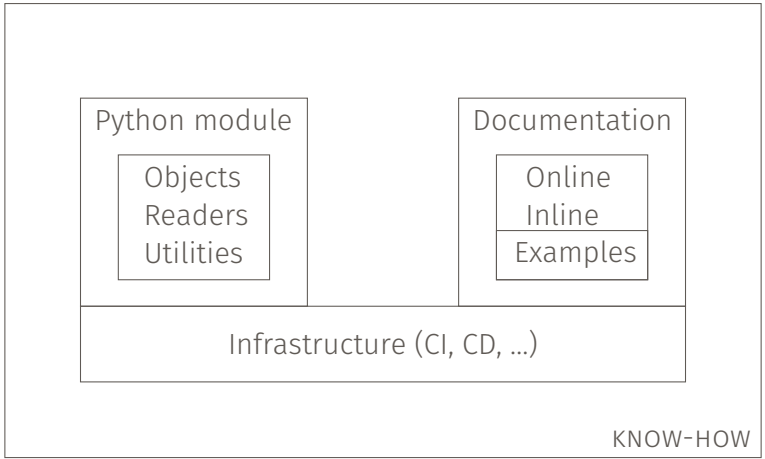
`kuibit` is workflow independent

STRUCTURE OF THE `KUIBIT` PROJECT



`kuibit` is workflow independent

STRUCTURE OF THE `KUIBIT` PROJECT



`kuibit` is workflow independent

KUIBIT HAS THREE GROUPS OF MODULES

Objects

TimeSeries
FrequencySeries
UniformGridData
HierarchicalGridData
…

KUIBIT HAS THREE GROUPS OF MODULES

Objects

- TimeSeries
- FrequencySeries
- UniformGridData
- HierarchicalGridData
- ...

Readers

- SimDir
- HorizonsDir
- MultipolesDir
- GravitationalWavesDir
- ScalarsDir
- GridFunctionsDir
- ...

KUIBIT HAS THREE GROUPS OF MODULES

Objects

TimeSeries
FrequencySeries
UniformGridData
HierarchicalGridData
…

Readers

SimDir
HorizonsDir
MultipolesDir
GravitationalWavesDir
ScalarsDir
GridFunctionsDir
…

Utilities

gw_mismatch
sYlm
sensitivity_curves
plot_color
…

UTILITIES

Convenience functions and useful routines:

- → `gw_utils` (e.g., `luminosity_distance_to_redshift`, `antenna_pattern`)
- → `unitconv` (e.g., from geometrized to physical and viceversa)
- → `gw_mismatch`
- → `sensitivity_curves` (LISA, aLIGO, CE, ET, ...)

Helpers for command-line scripts and visualization

- → `argparse_helper`
- → `visualize_matplotlib`

OBJECTS (TIME AND FREQUENCY SERIES AND GRID DATA)

→ High-level abstractions for data and useful methods

→ E.g., `TimeSeries` (`ts(10)`, `ts1 + np.sin(ts2)`)

OBJECTS (TIME AND FREQUENCY SERIES AND GRID DATA)

- → High-level abstractions for data and useful methods
- → E.g., `TimeSeries` (`ts(10)`, `ts1 + np.sin(ts2)`)
- → `UniformGridData` for data on a regular patch
- → `HierarchicalGridData` is essentially a collection of `UniformGridData`
- → `HierarchicalGridData` cannot be visualized directly and have to be resampled to `UniformGridData`

READERS DEAL WITH THE OUTPUT MESS AND PRESENT US WITH AN OBJECT

Readers:

- → Find the files associated to what you asked
- → Deal with reading (e.g., HDF5 files, compressed files, reading correct column)
- → Clean up the data (e.g., simulation restarts)

`SimDir`	Main point of entry (find all the files)
`*Dir (e.g., GridFunctionsDir)`	Process files from `SimDir`
`All* (e.g., AllGridFunctions)`	Organizes in the various variables
`One* (e.g., OneGridFunction)`	Has one variable (usually indexed by iterations)

All are dictionary-like that you can print, or get keys, or access with attributes.

CALL FOR CONTRIBUTIONS

- → `kuibit` is a great framework to make your codes available to the entire community
- → Openly developed, accessible, well-commented, easy-to-extend
- → Great learning opportunity!
- → **github.com/Sbozzolo/kuibit**

- → `kuibit` is published in the Journal of Open-Source Software
- → Telegram user group/support/announcements at **t.me/kuibit**
- → Feel free to reach me at `gabrielebozzola@email.arizona.edu`
- → A *kuibit* is a Tohono O'odham stick to pluck Saguaro's fruit

Harvest the fruit of your `Cactus` simulations with `kuibit`!

Credits: Wolfgang Kastaun, NumPy, SciPy, h5py, TACC, NSF, NASA

SETUP

https://sbozzolo.github.io/kuibit/first_steps.html
https://sbozzolo.github.io/kuibit/recommendation_examples.html