Balorog

For those who dig too deeply and greedily into their data...

Eric Suchyta and Eric Huff

Contents

1 Introduction

Balrog is ...

2 Usage

Usage

3 Command Line Options

Balrog runs can be configured via command line options. Two types of options exist. First are the built-in ones, native to Balrog. In addition, Balrog supports a mechanism for users to define their own command line options. To print a list of all Balrog's command line options, both native and user-defined, along with brief help strings, run:

```
% ./balrog.py --help
```

Section 3.1 further details each of the native options and Section 3.2 explains how to create custom options.

3.1 Built-in Options

Option Name	Description
`--outdir`	Toplevel directory for Balrog output files. Files will be organized into intuitively named directories under `--outdir`. **DEFAULT**: `$INSTALLDIR/default_example/output/`
`--imagein`	Image to insert simulated galaxies into. Must be in FITS format. **DEFAULT**: `$INSTALLDIR/default_example/input/example.fits`
`--imageext`	Index of the FITS extension where the image flux data lives. Indexing begins at 0. **DEFAULT**: 0
`--weightin`	File containing the weight map associated with `--imagein`. This can be a separate file from `--imagein` or the same file, where the flux image and weigh map live in different extensions. **DEFAULT**: `$INSTALLDIR/default_example/input/example.fits`
`--weightext`	Index of the FITS extension where the weight map data lives. Indexing begins at 0.

DEFAULT:

```
if --imagein != --weightin:
  --weighext = 0
else:
  --weightext = --imageext + 1
```

`--psfin`

File containing the PSFEx PSF model for `--imagein`. This is a FITS file, but the convention uses `.psf` as the extension.
DEFAULT: `$INSTALLDIR/default_example/input/example.psf`

`--xmin`

x-coordinate pixel for the lower bound of the subimage (if subsampling). $x \in [1, N_{\mathrm{cols}}]$.
DEFAULT: 1

`--xmax`

x-coordinate pixel for the upper bound of the subimage (if subsampling). $x \in [1, N_{\mathrm{cols}}]$.
DEFAULT: N_{cols}

`--ymin`

y-coordinate pixel for the lower bound of the subimage (if subsampling). $y \in [1, N_{\mathrm{rows}}]$.
DEFAULT: 1

`--ymax`

y-coordinate pixel for the upper bound of the subimage (if subsampling). $y \in [1, N_{\mathrm{rows}}]$.
DEFAULT: N_{rows}

`--ngal`

Number of galaxies to simulate.
DEFAULT: 40

`--gain`

Gain [e^{-}/ADU] for adding CCD noise to the simulated galaxies. Refer to galsim.CCDNoise documentation for further details. `--gain` can take two types of values: a float explicitly defining the gain, or a string referring to a keword written in the header of `--imagein[--imageext]`. If neither of these is successfully found, Balrog uses the defaults.
DEFAULT:

```
try:
  --gain = --imagein[--imageext].header['GAIN']
except:
  --gain = 1.0
```

`--zeropoint`

Zeropoint for converting sampled simulation magnitudes to simulated fluxes. SExtractor will run using this zeropoint when reporting magnitudes. `--zeropoint` can take two types values: a float explicitly defining the zeropoint, or a string referring to a keword written in the header of `--imagein[--imageext]`. If neither of these is successfully found, Balrog uses the defaults.

DEFAULT:

```
try:
   --zeropoint = --imagein[--imageext].header['SEXMGZPT']
except:
   --zeropoint = 30.0
```

`--seed`

Seed to give random number generator for any sampling which requires it, except noise realizations which are always different.

DEFAULT: Current time

`--fluxthresh`

Flux threshold below which the simulated galaxy's profile must fall before drawing the postage stamp.

DEFAULT: 0.01

`--clean`

Delete Image files after catalogs have been written.

DEFAULT: Unflagged, i.e. effectively false

`--sexpath`

Full path to SExtractor executable.

DEFAULT: sex, i.e. system default

`--sexconfig`

Configuration file for running SExtractor. Refer to the SExtractor user manual or Source Extractor for Dummies for more help.

DEFAULT: `$INSTALLDIR/astro_config/sex.config`

`--sexparam`

Parameter file specifying which measurements SExtractor outputs. Refer to the SExtractor user manual or Source Extractor for Dummies for more help.

DEFAULT: `$INSTALLDIR/astro_config/bulge.param`. This performs Sérsic profile model fits to each galaxy with free Sérsic index.

`--sexnnw`

SExtractor neural network file for star-galaxy separation. Refer to the SExtractor user manual or Source Extractor for Dummies for more help.

DEFAULT: `$INSTALLDIR/astro_config/sex.nnw`

`--sexconv` SExtractor filter convolution file when making detections. Refer to the SExtractor user manual or Source Extractor for Dummies for more help.
DEFAULT:

`--noassoc` Do not run SExtractor in association mode. Association mode is SExtractor speak to only look for sources at certain positions; here, the simulated galaxy positions. Using association mode is signficantly faster when simulating into an image consisting of many objects prior to any simulation.
DEFAULT: Unflagged, i.e. use association mode.

`--noempty` Skip SExtractor run over original image, prior to any simulation. One usage for such a run is to identify cases where a galaxy is simulated in the same position as something originally there. Depending on how the objects' properties conspire, SExtractor may not know any blending happened.
DEFAULT: Unflagged, i.e. perform the SExtractor run

`--sexemptyparam` Parameter file specifying which measurements SExtractor outputs during run over original image, prior to any simulation. If only interested in the run for 'deblending' issues, the file's contents are mostly irrelevant. The default file does not do model fitting to be faster.
DEFAULT: `$INSTALLDIR/astro_config/sex.param`

3.2 User-defined Options

Within the `config.py` file, user's are able to define their own command line options. This occurs within the function `CustomArgs`. Passed to `CustomArgs` as an argument is `parser`, an object made by `python`'s `argparse.ArgumentParser()`. Arguments can be added to parser according to the usual `argparse` syntax. For those unfamilar with `argparse`, this tutorial contains many useful examples. A simple example of `CustomArgs` is copied below.

```
def CustomArgs(parser):
    parser.add_argument( "-cs", "--catalogsample", help="Catalog used to
                         sample simulated galaxy parameter distriubtions
                         from", type=str, default=None)
```

User-defined options are parsed within the function `CustomParseArgs`, also part of `config.py`. Passed as an argument to `CustomParseArgs` is `args`, equivalent to an object returned by `parser.parse_args()`. Each one of the user's command line options becomes an attribute of `args`. A simple version of `CustomParseArgs` has been included below.

```
def CustomParseArgs(args):
    thisdir = os.path.dirname( os.path.realpath(__file__) )
    if args.catalogsample==None:
       args.catalogsample = os.path.join(thisdir, 'cosmos.fits')
```

The ability to define and parse one's own command line arguments is intended to make Balrog flexible to conveniently running a wide variety of different simulation scenarios.

4 Defining the Simulation

Something

5 Output

Each Balrog run generates a number of output files. These are organized into a fixed directory structure. Users indicate the `--outdir` command line option, and the remainder of the naming scheme occurs automatically, placing files in subdirectories under `--outdir`. Four subdirectories are written, labelled accoring to what type of files they contain.

- `balrog_cat`: Contains catalog files.
- `balorg_image`: Contains image files.
- `balrog_log`: Contains log files.
- `balrog_sexconfig`: Contains files for configuring sextractor.