

BARTPy: Exposing the BART Library to Python

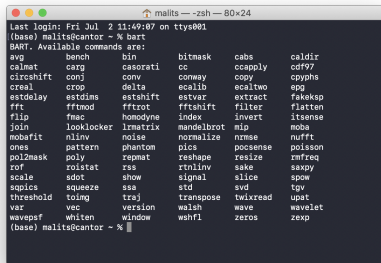
Max Litster

UC Berkeley Dept. of EECS

09/21/2021

Introduction

- ▶ BART: High-Performance CLI Tools for Computational MRI
 - ▶ Command-Line Tools
 - ▶ Low-Level Libraries
- ▶ Extensive network of scientific Python libraries
- ▶ Exposed to Python, through a command-line wrapper

A terminal window titled 'malits -- zsh -- 80x24' showing the output of the 'bart' command. The output lists 60 available command-line tools arranged in a grid. The tools include various image processing and analysis utilities like 'avg', 'calmat', 'circularshift', etc. The prompt is '(base) malits@cantor ~ %' and the cursor is at the end of the line.

```
Last login: Fri Jul 2 11:49:07 on ttys001
(base) malits@cantor ~ % bart
BART. Available commands are:
avg      bonch      bin          bitmask     cabs         caldir
calmat   carg         casorati    cc          ccapply     cdf97
circularshift conj      conv        conway      copy        cpyphs
creal    crop         delta       ecalib      ecaltwo     epg
estdelay estdms      estshift    estvar      extract     fakeksp
fft       fftmod      fftrot      fftshift    filter       flatten
flip     fmac        homodyne    index       invert      itsense
join     locklocker  lsmatrix    mandelbrot  mip         moba
mobafit  nlinv       noise       normalize   nmse        nufft
ones     pattern     phantom     pics        pocsense    poisson
pol2mask poly        repmat      reshape     resize      rmfreq
rof       roistat     rss         rtnlinv     sake        saxpy
scale     sdot        show        signal      slice       spow
sqpics   squeeze     ssa         std         svd         tgv
threshold toimg      traj        transpose   twixread    upat
var       vec         version     walsh       wave        wavelet
wavepsf   whiten      window     wshfl      zeros       zexp
(base) malits@cantor ~ %
```

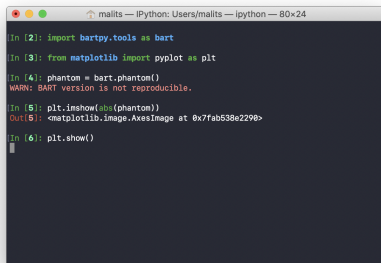
Figure 1: CLI Tools

Goals

- ▶ New Python interface, more intuitive
- ▶ Easily interoperable with Numpy, Scipy, etc.
- ▶ Minimal overhead (automated)
- ▶ Expose low-level libraries for developing tools in Python

Current Release

- ▶ Automated generation of functions for CLI tools
- ▶ For writing high-level recon pipelines in Python
- ▶ Low-level bindings in experimental phase

A screenshot of an IPython terminal window. The title bar at the top reads "malits — IPython: Users/malits — ipython — 80x24". The terminal has a dark background with light green text. It shows a series of input and output prompts. The inputs are: "In [2]: import bartpy.tools as bart", "In [3]: from matplotlib import pyplot as plt", "In [4]: phantom = bart.phantom()", "In [5]: plt.imshow(phantom)", and "In [6]: plt.show()". The output for the fifth input is "Out[5]: <matplotlib.image.AxesImage at 0x7fab538e2290>". A warning message "WARN: BART version is not reproducible." is displayed between the fourth and fifth inputs. The cursor is at the end of the sixth input line.

```
malits — IPython: Users/malits — ipython — 80x24

In [2]: import bartpy.tools as bart

In [3]: from matplotlib import pyplot as plt

In [4]: phantom = bart.phantom()
WARN: BART version is not reproducible.

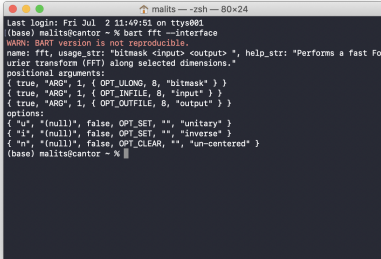
In [5]: plt.imshow(phantom)
Out[5]: <matplotlib.image.AxesImage at 0x7fab538e2290>

In [6]: plt.show()
```

Figure 2: Python Code Example

Requirements

- ▶ Latest release of BART from Github

A terminal window titled 'malits -- zsh -- 80x24' showing the output of the 'bart fft --interface' command. The output includes the BART version (0.9.0), a warning that the version is not reproducible, and the usage information for the 'fft' command, including positional arguments and options.

```
Last login: Fri Jul 2 11:49:51 on ttys001
(base) malits@cantor ~ % bart fft --interface
WARN: BART version is not reproducible.
name: fft, usage_str: "bitmask <input> <output> ", help_str: "Performs a fast Fourier transform (FFT) along selected dimensions."
positional arguments:
  { true, "ARG", 1, { OPT_ULONG, 8, "bitmask" } }
  { true, "ARG", 1, { OPT_INFILE, 8, "input" } }
  { true, "ARG", 1, { OPT_OUTFILE, 8, "output" } }
options:
  { "u", "(null)", false, OPT_SET, "", "unitary" }
  { "i", "(null)", false, OPT_SET, "", "inverse" }
  { "n", "(null)", false, OPT_CLEAR, "", "un-centered" }
(base) malits@cantor ~ %
```

Figure 3: Interface feature

Looking to get Involved?

Seeking developers to help develop internal bindings and test existing code. Github repo (linked) has example code and a Developer guide for open-source developers looking to contribute.

- ▶ Testing
- ▶ Coverage
- ▶ Automation Tools (SWIG Typemaps)