amber_meta Documentation

Release 0.0.1

D. Vohl

Jan 30, 2020

Contents:

This repository integrates a few routines to launch amber in a systematic manner.

CHAPTER 1

Getting the code

```
git clone https://github.com/macrocosme/amber_meta.git
cd amber_meta/
pip[3] install -r requirements.txt
```

CHAPTER 2

Requirements

```
## These can't be pip installed
# sigproc
# filterbank
# http://github.com/liamconnor/arts-analysis

# pip[3] install -r requirements.txt
PyYAML>=3.13
matplotlib>=3.0.3
pandas>=0.21.1
seaborn>=0.9.0
sphinx_automodapi>=0.10
```

CHAPTER 3

Usage

The most basic usage is via *python amber_run.py*, and parameters that will be prompted.

Else, more advanced usage involves functions not yet added to amber_runs's main. In an ipython session:

```
import amber_meta.amber_run as ar
import amber_meta.amber_plot as ap

# Run amber using root scenario yaml file
'''
The amber job(s) will run independently. The following steps currently
involves that these jobs have terminated and their .trigger outputs
be available.
'''
imput_file = 'yaml/root/root.yaml'
ar.run_amber_from_yaml_root(
  input_file,
  root='subband',
  verbose=False,
  print_only=True
) # Print only will not launch the amber job. When False, the command will be run via␣
↪subprocess.

# Read amber output .trigger files (e.g. steps 1..N) pooled into a pandas dataframe
df = ar.get_amber_run_results_from_root_yaml(
  input_file,
  root='subband',
  verbose=False
)

# Make pair plot from output
pairplot(
  df,
  output_name='../pairplot.pdf'
)
```

CHAPTER 4

Example of root yaml file

```
# AMBER setup for bruteforce dedispersion
bruteforce:
    input_file: 'path/to/filterbank.fil'
    n_cpu: 1
    base_name: 'scenario_base_name'
    base_scenario_path: 'scenario/' # Path where amber scenario files live
    scenario_files:  ['tuning.sh']
    snrmin: 8
    base_config_path: 'configuration/' # Path where amber configuration files live
    config_repositories: ['scenario_base_name']
    debug: False
    rfim: True
    rfim_mode: 'time_domain_sigma_cut'
    rfim_threshold: None
    snr_mode: 'snr_mom_sigmacut'
    input_data_mode: 'sigproc'
    output_dir: 'results/'
    verbose: True
    print_only: False
# AMBER setup for subband dedispersion
subband:
    input_file: 'path/to/filterbank.fil'
    n_cpu: 3
    base_name: 'scenario_base_name'
    base_scenario_path: 'scenario/'
    scenario_files:  [
        'tuning_1.sh',
        'tuning_2.sh',
        'tuning_3.sh'
      ]
    snrmin: 8
    base_config_path: 'configuration/'
    config_repositories: [
        'scenario_base_name_step1',
```

(continues on next page)

(continued from previous page)

```
        'scenario_base_name_step2',
        'scenario_base_name_step3'
      ]
    debug: False
    rfim: True
    rfim_mode: 'time_domain_sigma_cut'
    rfim_threshold: None
    snr_mode: 'snr_mom_sigmacut'
    input_data_mode: 'sigproc'
    output_dir: 'results/'
    verbose: True
    print_only: False
```

CHAPTER 5

License

This project is licensed under the terms of the GNU GPL v3+ license.

5.1 amber_run

amber_meta.amber_run.**AMBER_SETUP_PATH = '/home/vohl/AMBER_setup/'**

amber_meta.amber_run.**create_amber_command**(*base_name='scenario_3_partitions'*, *input_file='data/filterbank/file.fil'*, *scenario_file='$SOURCE_ROOT/scenario/3_dms_partitions/scenario_3_p*
config_path='$SOURCE_ROOT/install/scenario_3_partitions_step1/', *rfim=True*, *rfim_mode='time_domain_sigma_cut'*, *rfim_threshold_tdsc=None*, *rfim_threshold_fdsc=None*, *snr_mode='snr_mom_sigmacut'*, *input_data_mode='sigproc'*, *cpu_id=1*, *snrmin=10*, *output_dir='$OUTPUT_ROOT/results/'*, *verbose=True*, *root_name=None*)

Launch amber.

Creates an amber launch command to be run with subprocess.

Parameters

- **base_name** (*str*) – Base name.
- **input_file** (*str*) – Intput filterbank file.
- **scenario_file** (*str*) – Scenario file (including path)
- **config_path** (*str*) – Path of configuration files
- **rfim** (*bool*) – Use RFI mitigation or not.

- **rfim_mode** (*str*) – RFI mitigation mode. Choices: [time_domain_sigma_cut | frequency_domain_sigma_cut]
- **rfim_threshold** (*str*) – Override rfim threshold value. Default: None
- **snr_mode** (*str*) – SNR mode. Choices: [snr_standard | snr_momad | snr_mom_sigmacut]
- **input_data_mode** (*str*) – Input data mode. Choices: [sigproc | data]
- **cpu_id** (*int*) – CPU id for process and GPU.
- **snrmin** (*int*) – Minimum SNR for outlier detection.
- **output_dir** (*str*) – Output directory.
- **verbose** (*bool*) – Print extra information at runtime.
- **root_name** (*str*) – Root name used for output.

amber_meta.amber_run.**create_rfim_configuration_threshold_from_yaml_root**(*input_yaml_file*, *root='subband'*, *rfim_threshold_tdsc='3.25'* *rfim_threshold_fdsc='2.50'* *verbose=False*, *print_only=False*)

Create RFIm configuration file starting from with a yaml root

Parameters

- **input_yaml_file** (*str*) – Input root yaml file
- **root** (*str*) – Root value of the yaml file. Default: 'subband'
- **threshold** (*list*) – New threshold file to be generated. Default: '2.50',
- **verbose** (*bool*) – Print extra information at runtime. Default: False.
- **print_only** (*bool*) – Only print command, do not launch them. Default: False.

amber_meta.amber_run.**create_rfim_configuration_thresholds_from_yaml_root**(*input_yaml_file, root='subband', thresholds=['2.00', '2.50', '3.00', '3.50', '4.00', '4.50', '5.00'], verbose=False, print_only=False*)

Create RFIm configuration files starting from with a yaml root

Parameters

- **input_yaml_file** (*str*) – Input root yaml file
- **root** (*str*) – Root value of the yaml file. Default: 'subband'
- **thresholds** (*list*) – Thresholds files to be generated. Default: ['2.00', '2.50', '3.00', '3.50', '4.00', '4.50', '5.00'],

- **verbose** (*bool*) – Print extra information at runtime. Default: False.
- **print_only** (*bool*) – Only print command, do not launch them. Default: False.

amber_meta.amber_run.**get_amber_run_results_from_root_yaml**(*input_yaml_file*, *root='subband'*, *verbose=False*)

Run amber starting from a yaml root scenario file.

Launches a amber scenario where each step is run as independent sub-processes.

Parameters

- **input_yaml_file** (*str*) – Accepted format are .yaml and .yml
- **root** (*str*) – Name of root scenario in input yaml.
- **verbose** (*bool*) – Print extra information at runtime.

amber_meta.amber_run.**run_amber_from_yaml_root**(*input_yaml_file*, *root='subband'*, *rfim_threshold_override=False*, *rfim_threshold_tdsc='3.25'*, *rfim_threshold_fdsc='2.50'*, *verbose=False*, *print_only=True*, *detach_completely=True*)

Run amber starting from a yaml root scenario file.

Launches a amber scenario where each step is run as independent sub-processes.

Parameters

- **input_yaml_file** (*str*) – Input filename with .yaml or .yml extension.
- **root** (*str*) – Name of root scenario in input yaml.
- **verbose** (*bool*) – Print extra information at runtime.
- **print_only** (*bool*) – Only print command, do not launch them.
- **detach_completely** (*bool*) – If True, launch all processes and detach from them. Else, wait on last cpu.

amber_meta.amber_run.**run_amber_from_yaml_root_override_threshold**(*input_basename='yaml/root/root'*, *root='subband'*, *threshold='2.00'*, *verbose=False*, *print_only=False*, *detach_completely=True*)

Run amber from a yaml root file and override threshold for RFIm

input_basename [str] Default: 'yaml/root/root'

root [str] Default: 'subband',

threshold [str] Default: '2.00'

verbose [bool] Print extra information at runtime. Default: False.

print_only [bool] Only print command, do not launch them. Default: False.

amber_meta.amber_run.**run_amber_from_yaml_root_override_thresholds**(*input_basename='yaml/root/root', root='subband', thresholds_tdsc=['3.25'], thresholds_fdsc=['2.00', '2.25', '2.50', '2.698', '2.75'], verbose=False, print_only=False, detach_completely=False*)

Run amber from a yaml root file and for multiple overriden threshold for RFIm

input_basename [str] Default: 'yaml/root/root'

root [str] Default: 'subband',

thresholds [list] Default: ['2.00', '2.50', '3.00', '3.50', '4.00', '4.50', '5.00']

verbose [bool] Print extra information at runtime. Default: False.

print_only [bool] Only print command, do not launch them. Default: False.

amber_meta.amber_run.**test_amber_run**(*input_file='data/dm100.0_nfrb500_1536_sec_20190214-1542.fil', n_cpu=3, base_name='tuning_halfrate_3GPU_goodcentralfreq', base_scenario_path='/home/vohl/software/AMBER/scenario/', scenario_files=['tuning_1.sh', 'tuning_2.sh', 'tuning_3.sh'], snrmin=8, base_config_path='$SOURCE_ROOT/configuration/', config_repositories=['tuning_halfrate_3GPU_goodcentralfreq_step1', 'tuning_halfrate_3GPU_goodcentralfreq_step2', 'tuning_halfrate_3GPU_goodcentralfreq_step3'], rfim=True, rfim_mode='time_domain_sigma_cut', snr_mode='snr_mom_sigmacut', input_data_mode='sigproc', verbose=True, print_only=False*)

Test amber.

Creates three amber jobs.

Parameters

- **amber_mode** (*str*) –
- **input_file** (*str*) –
- **n_cpu** (*int*) –
- **base_name** (*str*) –
- **base_scenario_path** (*str*) –
- **scenario_files** (*list*) –
- **snrmin** (*int*) –
- **base_config_path** (*str*) –
- **config_repositories** (*list*) –

- **rfim** (*bool*) –
- **rfim_mode** (*str*) –
- **snr_mode** (*str*) –
- **input_data_mode** (*str*) –
- **verbose** (*bool*) – Print extra information at runtime.
- **print_only** (*bool*) – Only print the command without launching it.

amber_meta.amber_run.**test_tune**(*base_scenario_path='/home/vohl/software/AMBER/scenario/', base_name='tuning_halfrate_3GPU_goodcentralfreq', scenario_files=['tuning_1.sh', 'tuning_2.sh', 'tuning_3.sh'], config_path='/home/vohl/software/AMBER/configuration/', verbose=True, print_only=True*)

Test tuning amber.

Launch tune_amber for three scenarios.

Parameters base_scenario_path : str base_name : str scenario_files : list config_path : str

amber_meta.amber_run.**tune_amber**(*scenario_file='/home/vohl/software/AMBER/scenario/tuning_step1.sh', config_path='/home/vohl/software/AMBER/configuration/tuning_step1', verbose=True, print_only=True*)

Tune amber.

Tune amber based on a scenario file. The output is save to config_path.

Parameters

- **scenario_file** (*str*) –
- **config_path** (*str*) –

5.2 amber_utils

amber_meta.amber_utils.**check_directory_exists**(*directory*)

Check if directory (string) ends with a slash.

If directory does not end with a slash, add one at the end.

Parameters **directory** (*str*) –

Returns **directory**

Return type str

amber_meta.amber_utils.**check_file_exists**(*file*)

Check if a file exists

file [str] Filename with path.

Returns **response** – Response to the question "does the file exist?".

Return type bool

amber_meta.amber_utils.**check_path_ends_with_slash**(*path*)

Check if directory (string) ends with a slash.

If directory does not end with a slash, add one at the end.

Parameters **directory** (*str*) –

Returns directory

Return type str

amber_meta.amber_utils.**create_rfim_configuration_thresholds**(*config_path*, *rfim_mode='time_domain_sigma_cut'*, *original_threshold_tdsc='2.50'*, *original_threshold_fdsc='2.50'*, *new_threshold_tdsc='3.25'*, *new_threshold_fdsc='2.50'*, *duplicate=True*, *verbose=False*, *print_only=False*)

Create a new RFIm configuration file for specified threshold

Parameters

- **config_path** (*str*) – Path to configuration files
- **rfim_mode** (*str (optional)*) – RFIm mode of operation. Default: 'time_domain_sigma_cut'
- **original_threshold** (*str (optional)*) – Threshold listed in base config file. Default: 2.50
- **new_threshold** (*str (optional)*) – New threshold. Default: 1.00
- **duplicate** (*bool*) – When True, make copies of the base configuration files adding the threshold in new filename
- **verbose** (*bool*) – Print extra information at run-time.
- **print_only** (*bool*) – Only print verbose information without running anything else.

amber_meta.amber_utils.**duplicate_config_file**(*config_path*, *base_filename*, *copy_filename*)

Duplicate a configuration file using copu_filename as output nameself.

Parameters

- **config_path** (*str*) – Path to configuration files
- **base_filename** (*str*) – Filename of file to be copied
- **copy_filename** (*str*) – Filename of duplicate

amber_meta.amber_utils.**find_replace**(*filename*, *text_to_search*, *text_to_replace*, *inplace=True*, *verbose=False*)

Find text_to_search in filename and replace it with text_to_replace

Parameters

- **filename** (*str*) – Filename of input file to modify
- **text_to_search** (*str*) – Text string to be searched in intput file
- **text_to_replace** (*str*) – Text string to replace text_to_search with in intput file
- **inplace** (*bool*) – Default: True

amber_meta.amber_utils.**get_filterbank_header**(*input_file*, *verbose=False*)

Get header and header_size from filterbank.

Parameters

- **input_file** (*str*) – Input filterbank file
- **verbose** (*bool*) – Print extra information at run-time.

Returns

- **header** (*dict*) – filterbank.read_header.header
- **header_size** (*int*) – filterbank.read_header.header_size

amber_meta.amber_utils.**get_full_output_path_and_file**(*output_dir*, *base_name*, *root_name=None*, *cpu_id=None*)

Get full output path and file name.

Parameters

- **output_dir** (*str*) –
- **base_name** (*str*) –
- **root_name** (*str*) –

Returns **path_and_file**

Return type str

amber_meta.amber_utils.**get_list_as_str**(*command*)

Turn command list to pretty print.

Prints each element of the 'command' list as a string.

Parameters **command** (*list*) –

Returns **c** – Prettified command

Return type str

amber_meta.amber_utils.**get_max_dm**(*scenario_dict*)

Compute maximum dm.

Parameters **scenario_dict** (*dict*) – Scenario dictionary outputed by amber_utils.parse_scenario_to_dictionary()

Returns **max_dm** – Maximum DM

Return type float

amber_meta.amber_utils.**get_nbatch**(*input_file*, *header*, *header_size*, *samples*, *verbose=False*)

Get number of batches (nbatch) available in filterbank

Parameters

- **input_file** (*str*) –
- **header** (*dict*) – filterbank.read_header.header
- **header_size** (*int*) – filterbank.read_header.header_size

Returns **nbatch**

Return type int

amber_meta.amber_utils.**get_root_name**(*input_file*)

Get yaml file's root name.

Parameters **input_file** (*str*) – Yaml input file

Returns **root_name**

Return type str

amber_meta.amber_utils.**get_scenario_file_from_root_yaml_base_dict**(*base*, *cpu_id=0*)

Get the scenario path and file from info in root yaml file.

Parameters

- **base** (*dict*) – Base dictionary as fetched from parse_scenario_to_dictionary
- **cpu_id** (*int*) – Index of the step

Returns

- **scenario_file** (*str*)
- *Usage*
- ——
- >>> *input_yaml_file = 'yaml/root/root.yaml'*
- >>> *root='subband'*
- >>> *base = parse_scenario_to_dictionary(input_yaml_file)[root]*
- >>> *scenario_file = get_scenario_file_from_root_yaml_base_dict(base, cpu_id=0)*

amber_meta.amber_utils.**list_files_in_current_path**(*path*, *extensions=None*)

Returns files in the current folder only

Parameters

- **path** (*str*) – Path from where to list files
- **extensions** (*list*) – List of desired extensions to include. Default: None. Usage example: ['.txt', '.trigger']

Returns **files**

Return type list

amber_meta.amber_utils.**list_files_with_paths_recursively**(*my_path*)

Recursively list files in my_path

Recursively list files in my_path and returns the list in the form of ['path/to/file/myfile.extension', '...']

Parameters **my_path** (*str*) –

amber_meta.amber_utils.**parse_scenario_to_dictionary**(*scenario_file*)

Parse an amber scenario file to a python dictionary

Accepted file extensions: [.yaml | .yml], and [.sh] as described in https://github.com/AA-ALERT/AMBER_setup/blob/development/examples/scenario.sh

Parameters **scenario_file** (*str*) – amber scenario file (including path)

Returns **scenario_dict** – parsed dictionary

Return type dict

amber_meta.amber_utils.**parse_sh_scenario_to_dictionary**(*scenario_file*)

Parse an amber scenario file to a python dictionary

File extension expected is '.sh' as described in https://github.com/AA-ALERT/AMBER_setup/blob/development/examples/scenario.sh

Note that the extension is not required per se, but the file structure should follow a shell variable structure.

Parameters **scenario_file** (*str*) – amber scenario file (including path)

Returns **scenario_dict** – parsed dictionary

Return type dict

amber_meta.amber_utils.**parse_yaml_scenario_to_dictionary**(*scenario_file*, *scenario_name=None*)

Parse an amber scenario file (yaml) to a python dictionary

Parameters **scenario_file** (*str*) – amber scenario file in yaml format (including path)

Returns **scenario_dict** – parsed dictionary

Return type dict

amber_meta.amber_utils.**pretty_print_command**(*command*)

Pretty print an amber command.

Prints each element of the 'command' list as a string.

Parameters **command** (*list*) –

5.3 amber_options

class amber_meta.amber_options.**AmberOptions**(*rfim=True*, *rfim_mode='time_domain_sigma_cut'*, *snr_mode='snr_mom_sigmacut'*, *input_data_mode='sigproc'*, *downsampling=False*)

Class representing amber's command line options.

The class can be instanciated using default values, or by passing parameters as input. All command options will be availble via self.options.

```
>>> amber_options = AmberOptions(rfim=False, snr_mode='snr_mom_sigmacut', input_
↪data_mode='sigproc', downsampling=False)
>>> amber_options.options
['print',
 'opencl_platform',
 'opencl_device',
 'device_name',
 'sync',
 'padding_file',
 'zapped_channels',
 'integration_steps',
 'integration_file',
 'compact_results',
 'output',
 'dms',
 'dm_first',
 'dm_step',
 'threshold',
 'snr_mom_sigmacut',
 'max_std_file',
 'mom_stepone_file',
 'mom_steptwo_file',
 'sigproc',
 'stream',
 'header',
```

(continues on next page)

(continued from previous page)

```
'data',
'batches',
'channels',
'min_freq',
'channel_bandwidth',
'samples',
'sampling_time',
'subband_dedispersion',
'dedispersion_stepone_file',
'dedispersion_steptwo_file',
'subbands',
'subbanding_dms',
'subbanding_dm_first',
'subbanding_dm_step']
```

Parameters

- **rfim** (*bool (optional)*) – Default: True
- **rfim_mode** (*str (optional)*) – RFIm mode of operation. Default: 'time_domain_sigma_cut'
- **snr_mode** (*str (optional)*) – SNR mode of operation. Default: 'snr_mom_sigmacut'
- **input_data_mode** (*str (optional)*) – Input data mode (sigproc's filterbank file or dada ringbuffer). Default: 'sigproc'
- **downsampling** (*bool (optional)*) – Enable downsampling. Default: False

options_base
List of basic options

Type list

options_tdsc
List of options for RFIm's time domain sigma cut

Type list

options_fdsc
List of options for RFIm's frequency domain sigma cut

Type list

options_rfim
Options to choose between RFIm modes

Type dict

options_snr_standard
List of options for SNR standard

Type list

options_snr_momad
List of options for SNR median of medians maximum absolute deviation

Type list

options_snr_mom_sigmacut
List of options for SNR median of medians sigma cut

Type list

options_SNR
Options to choose between SNR modes

Type dict

options_downsampling
List of options for downsampling

Type list

options_subband_dedispersion
List of options for subband dedispersion

Type list

options_sigproc
List of options for sigproc data input

Type list

options_dada
List of options for dada ringbuffer input

Type list

options_input_data
Options to choose between input data modes

Type dict

5.4 amber_configuration

class amber_meta.amber_configuration.**AmberConfiguration**(*rfim=False*, *rfim_mode='time_domain_sigma_cut'*, *downsampling=False*)
Class representing amber's configuration files. The class can be instanciated using default values, or by passing parameters as input. All command options will be availble via self.options.

Parameters

- **rfim** (*bool (optional)*) – Default: True
- **rfim_mode** (*str (optional)*) – RFIm mode of operation. Default: 'time_domain_sigma_cut'

suffix
Suffix of configuration files (.conf)

Type str

configurations
Configuration built at initialisation

Type dict

rfim_config_tdsc_files
List of configuration file names for RFIm's time domain sigma cut

Type list

rfim_config_fdsc_files
List of configuration file names for RFIm's frequency domain sigma cut

Type list

rfim_config_files
Options to choose between RFIm modes

Type dict

downsampling_configuration
'downsampling'

Type str

integration_steps
'integration_steps'

Type str

zapped_channels
'zapped_channels'

Type str

5.5 amber_results

amber_meta.amber_results.**get_header**(*filename*, *sep=' '*)
Get filterbank's header.

Parameters

- **filename** (*str*) – filterbank file to read
- **sep** (*str*) – Separator

Returns **header** – Filterbank's header

Return type dict

amber_meta.amber_results.**read_amber_run_results**(*run_output_dir*, *extensions=['.trigger']*, *verbose=False*, *sep=' '*)
Read amber results from a run.

Parameters

- **run_output_dir** (*str*) – Path to output .trigger files
- **extensions** (*list*) – Desired extension(s) to include. Default: ['.trigger']
- **verbose** (*bool*) – Print developement information
- **sep** (*str*) – Separator

Returns **df** – All results in one dataframe.

Return type Pandas.DataFrame

amber_meta.amber_results.**read_injected_txt**(*injected_txt_dir*, *injected_txt_file*, *max_rows=None*)
Read amber results from a run.

Parameters

- **run_output_dir** (*str*) – Path to output .trigger files
- **extensions** (*list*) – Desired extension(s) to include. Default: ['.trigger']
- **verbose** (*bool*) – Print developement information
- **sep** (*str*) – Separator

Returns **df** – All results in one dataframe.

Return type Pandas.DataFrame

5.6 amber_plot

amber_meta.amber_plot.**pairplot**(*df*, *output_name='../pairplot.pdf'*)

Function to plot a graphical scatter plots

For each pair of columns in the dataframe, plot a scatter plots.

Parameters

- **df** (*pandas.DataFrame*) –
- **output_name** (*str*) – Filename of output [.pdf | .png]

Python Module Index

a

Index

P

R

S

T

Z