SEEK: Signal Extraction and Emission Kartographer Documentation

Release 0.1.0

Joel Akeret

Mar 06, 2017

Contents

1 Contents: **3**

1.1 Installation . . . 3
1.2 Usage . . . 3
1.3 RFI mitigation . . . 4
1.4 seek Package . . . 5
1.5 Contributing . . . 21
1.6 Credits . . . 22
1.7 History . . . 22

2 Feedback **23**

Python Module Index **25**

SEEK is a flexible and easy-to-extend data processing pipeline for single dish radio telescopes. It takes the observed (or simulated) TOD in the time-frequency domain as an input and processes it into *healpix*maps while applying calibration and automatically masking RFI. The data processing is parallelized using *ivy's* parallelization scheme.

The **SEEK** package has been developed at ETH Zurich in the Software Lab of the Cosmology Research Group of the ETH Institute of Astronomy.

The development is coordinated on GitHub and contributions are welcome. The documentation of **SEEK** is available at readthedocs.org .

CHAPTER 1

Contents:

Installation

The project is hosted on GitHub. Get a copy by running:

```
$ git clone https://github.com/cosmo-ethz/seek.git
```

Install the package like this:

```
$ cd seek
$ pip install -r requirements.txt
$ python setup.py install --user
```

Alternatively, if you want to develop new features:

```
$ cd seek
$ python setup.py develop --user
```

Usage

To use SEEK, you can download the example data that was collected at the Bleien Observatory:

```
$ wget -r -nH -np --cut-dirs=2 http://people.phys.ethz.ch/~ast/cosmo/bgs_example_data/
```

then you will be able to process the data with seek:

```
$ seek --file-prefix=./bgs_example_data --map-name=BGS_maps.hdf --verbose=True seek.
↪config.process_survey_fft
```

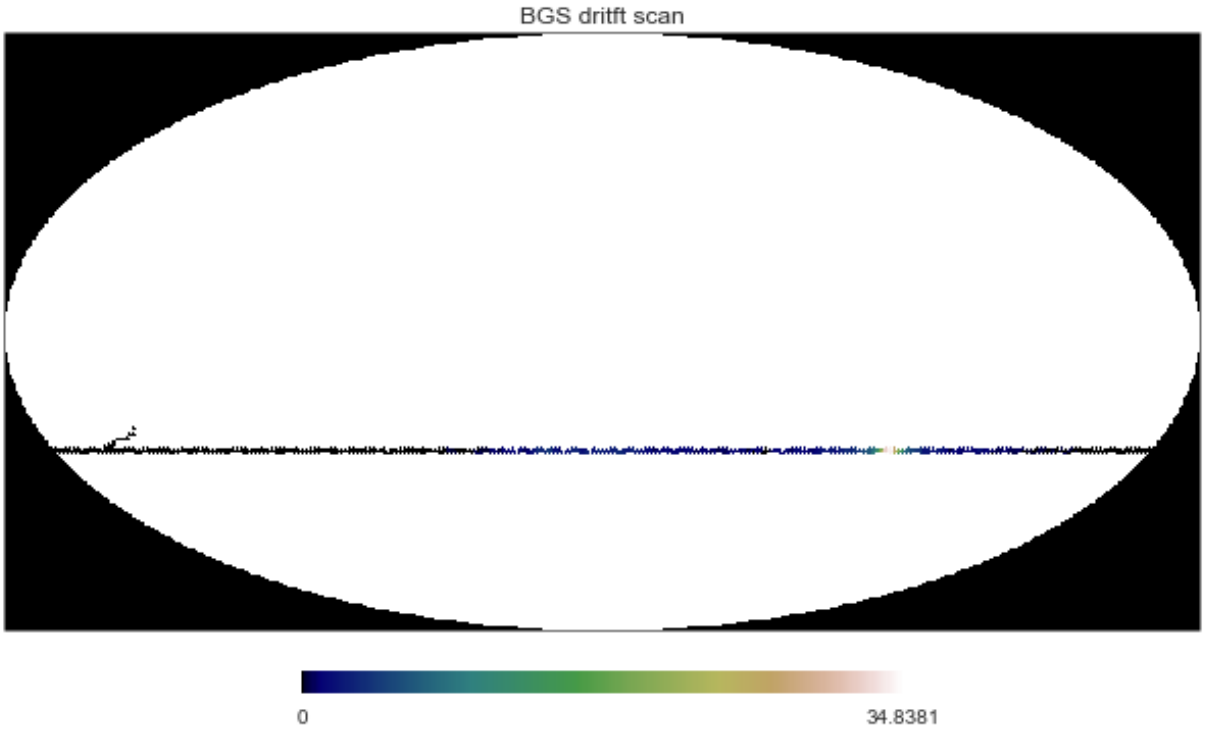
further option can be found in the *seek.config.process_survey_fft*

Finally, you can visualize the resulting map like this:

```
import h5py
import healpy as hp

with h5py.File("./BGS_maps.hdf", 'r') as fp:
    bgs_map = fp['MAPS'][20, 0]
    counts = fp['COUNTS'][20, 0]

bgs_map[counts==0] = hp.UNSEEN
hp.mollview(bgs_map, cmap="gist_earth")
```



RFI mitigation

SEEK's RFI mitigation follows the Offringa et al. *SumThreshold* algorithm. It's implemented in pure Python and JIT-compiled for speed with the HOPE package.

It can easily be used without of the SEEK data processing pipeline:

```
import numpy as np
from seek.mitigation import sum_threshold

rfi_mask = sum_threshold.get_rfi_mask(tod=np.ma.array(data),
                                      chi_1=20,
                                      sm_kwargs=sum_threshold.get_sm_kwargs(40, 20, 15, 7.
↪5),
                                      di_kwargs=sum_threshold.get_di_kwrags(3, 7))
```

The TOD has to be a Numpy masked array when passed to the *sum_threshold* algorithm. The other parameters are optional an give you the possiblity to tune the mitigation. Crucial is the a good starting value of *chi_1*, best explored

by trial and error. The keyword-arguments control the smoothing and dilation process. Further options can be found in the documentation of the module.

The resulting boolean mask looks something like that

seek Package

seek Package

class seek.__init__.**Coords**(*ra*, *dec*, *az*, *el*, *t*)
Bases: object

Subpackages

calibration Package

astro_calibration_source Module

Created on Aug 20, 2015

author: cchang

Models taken from: Baars 1997, Hafez 2008, Benz 2009 All numbers divided by 2 to account for the fact that our data is single polarization.

seek.calibration.astro_calibration_source.**barrs77_power_law**(*freq*, *a*, *b*, *c*)
Power law model from Baars 1997.

Parameters

- **freq** – frequency in Hz
- **a** – model parameter
- **b** – model parameter
- **c** – model parameter

Returns model spectra

seek.calibration.astro_calibration_source.**benz_sun**(*freq*)
Model of Solar spectra based on Benz 2009.

Parameters **freq** – frequency in Hz

Returns model spectra

seek.calibration.astro_calibration_source.**black_body**(*freq*, *T*)
Black body model spectra

Parameters

- **freq** – frequency in Hz
- **T** – temperature of black body

Returns model spectra

seek.calibration.astro_calibration_source.**source**(*obj*, *freq*)
Given name of calibration source and frequency grid, give model spectrum.

`fitting` Module

Created on Aug 18, 2015

author: cchang

`seek.calibration.fitting.`**`fit_func`**(*x*, *y*, *func_name*, *p0*)
Fit different functions to curve. The current choices for functions are single or double Gaussians plus a linear background.

`seek.calibration.fitting.`**`gauss2`**(*x*, *a_1*, *x0_1*, *sigma_1*, *a_2*, *x0_2*, *sigma_2*, *b*, *c*)
Double Gaussian model plus a linear background.

`flux_calibration_transit` Module

Created on Feb 4, 2016

author: cchang

class `seek.calibration.flux_calibration_transit.`**`CalibrationSource`**(*date*, *azimut*, *elevation*, *target*)

Bases: `tuple`

`azimut`
Alias for field number 1

`date`
Alias for field number 0

`elevation`
Alias for field number 2

`target`
Alias for field number 3

class `seek.calibration.flux_calibration_transit.`**`GaussFitResult`**(*gauss_A*, *gauss_x0*, *gauss_sigma*, *sky1*, *sky2*, *gauss_A_err*, *gauss_x0_err*, *gauss_sigma_err*, *sky1_err*, *sky2_err*, *rsquared*)

Bases: `tuple`

`gauss_A`
Alias for field number 0

`gauss_A_err`
Alias for field number 5

`gauss_sigma`
Alias for field number 2

`gauss_sigma_err`
Alias for field number 7

`gauss_x0`
Alias for field number 1

gauss_x0_err
Alias for field number 6

rsquared
Alias for field number 10

sky1
Alias for field number 3

sky1_err
Alias for field number 8

sky2
Alias for field number 4

sky2_err
Alias for field number 9

seek.calibration.flux_calibration_transit.**calibrate**(*ctx*, *plot=False*)
Main function that performs the calibration, including the following steps:

- loop through the calibration sources
  - find the corresponding list of files
  - —RFI removal
  - for each frequency fit 1D Gaussian to data
  - —divide amplitude by reference spectra
  - store the final pseudo-gain value
- take median pseudo-gain over all calibration sources
- fit this final spectra with a template derived from Sun
- store this final fit to memory

Parameters **ctx** – context that contains all parameters for RFI removal and file paths

Returns 2D array with first column frequency and second gain factor

seek.calibration.flux_calibration_transit.**fit_gaussian_source**(*data*, *time_axis*, *rfi_mask*, *source_dec*, *ctx*)
Given a chunck of 2D data, fit a 1D Gaussian to each frequency. Here only return the amplitude.

Parameters

- **data** – data
- **time_axis** – time axis
- **rfi_mask** – mask
- **source_dec** – declination coordinate of source
- **ctx** – context

Returns amplitude of Gaussian fit

seek.calibration.flux_calibration_transit.**fit_gaussian_source_full**(*data*, *time_axis*, *rfi_mask*, *source_dec*, *params*)

Given a chunck of 2D data, fit a 1D Gaussian to each frequency. Here return all fitted parameters.

Parameters

- **data** – data
- **time_axis** – time axis
- **rfi_mask** – mask
- **source_dec** – declination coordinate of source
- **params** – context parameters

Returns all Gaussian fit parameters

seek.calibration.flux_calibration_transit.**fit_sun_to_gain**(*gauss_amp*, *freq*, *params*)

Takes the array of Gaussian fit parameters and fits the sun template to it. Then calculate the actual gain that converts the map ADU to K.

seek.calibration.flux_calibration_transit.**process_calibration_transits**(*calibration_paths*, *ctx*)

Derive gain as a function of frequency for each transit measurement.

Parameters

- **calibration_paths** – path for calibration files
- **ctx** – context

Returns gain as a function of frequency

seek.calibration.flux_calibration_transit.**process_source**(*source*, *file_paths*, *gain_template*, *sm_kwargs*, *di_kwargs*, *ctx*)

Process a single transit measurement.

Parameters

- **source** – name of source
- **file_paths** – file path
- **gain_template** – template derived from the Sun
- **sm_kwargs** – SumThreshold parameters
- **di_kwargs** – SumThreshold parameters
- **ctx** – context

Returns parameters for Gaussian fit

config Package

common Module

Created on Jan 5, 2015

author: seehars

Common parameters that are always loaded.

process_survey Module

Created on Jan 23, 2015

author: seehars

Config file that specifies the Ivy workflow and other parameters used to run SEEK.

process_survey_fft Module

Created on Dec 18, 2015

author: jakeret

Config file that specifies parameters specific for the FFT spectrometer. Includes the other two config files to avoid duplication.

mapmaking Package

filter_mapper Module

Created on Feb 26, 2016

author: jakeret

seek.mapmaking.filter_mapper.**filter_data**(*data*)
remove outliers in data array.

Parameters **data** – data in the restructured form after create_maps.py

Returns data with outlier removed

seek.mapmaking.filter_mapper.**get_mapped_values**(*data*, *ctx*)
Maps the data by removing outliers and then computing the median per pixel. Follows http://stackoverflow.com/a/16562028/4067032 :param data: data in the restructured form after create_maps.py :param ctx: context

Returns median and sum of all the un-masked data in each healpix pixel

filter_variance_mapper Module

Created on Feb 26, 2016

author: jakeret

seek.mapmaking.filter_variance_mapper.**get_mapped_values**(*data*, *ctx*)
Maps the data by removing outliers and then computing the variance per pixel. :param data: data in the restructured form after create_maps.py :param ctx: context

Returns variance and sum of unmasked map

healpy_mapper Module

simple_mapper Module

Created on Feb 26, 2016

author: jakeret

seek.mapmaking.simple_mapper.**get_mapped_values**(*re_data*, *ctx*)
Maps the data by simply computing the mean per pixel :param re_data: data in the restructured form after create_maps.py

Returns mean and sum of unmasked data

simple_variance_mapper Module

Created on Feb 26, 2016

author: jakeret

seek.mapmaking.simple_variance_mapper.**get_mapped_values**(*re_data*, *ctx*)
Maps the data by simply computing the variance per pixel :param re_data: data in the restructured form after create_maps.py

Returns variance and sum of unmasked data

mitigation Package

outlier_masking Module

Created on Jan 22, 2015

author: seehars

seek.mitigation.outlier_masking.**getMask**(*tod*, *multiplicator*)
Construct outlier mask based on user-specified rejection criterion.

Parameters

- **tod** – TOD
- **multiplicator** – the number of standard deviation from the mean beyond which the data point is considered an outlier

Returns outlier mask

seek.mitigation.outlier_masking.**rm_rfi**(*ctx*)
Remove RFI by outlier rejection of a number of measurements in the same healpix pixel.

Parameters **ctx** – context

Returns outlier mask

sum_threshold Module

Created on Jan 21, 2015

author: jakeret

seek.mitigation.sum_threshold.**binary_mask_dilation**(*mask*, *struct_size_0*, *struct_size_1*)

Dilates the mask.

Parameters

- **mask** – original mask
- **struct_size_0** – dilation parameter
- **struct_size_1** – dilation parameter

Returns dilated mask

seek.mitigation.sum_threshold.**get_di_kwrags**(*struct_size_0=3*, *struct_size_1=3*)

Creates a dict with the dilation keywords.

Parameters

- **struct_size_0** – struct size in axis=0
- **struct_size_1** – struct size in axis=1

Returns dictionary with the dilation keywords

seek.mitigation.sum_threshold.**get_rfi_mask**(*tod*, *mask=None*, *chi_1=35000*, *eta_i=[0.5, 0.55, 0.62, 0.75, 1]*, *normalize_standing_waves=True*, *suppress_dilation=False*, *plotting=True*, *sm_kwargs=None*, *di_kwargs=None*)

Computes a mask to cover the RFI in a data set.

Parameters

- **data** – array containing the signal and RFI
- **mask** – the initial mask
- **chi_1** – First threshold
- **eta_i** – List of sensitivities
- **normalize_standing_waves** – whether to normalize standing waves
- **suppress_dilation** – if true, mask dilation is suppressed
- **plotting** – True if statistics plot should be displayed
- **sm_kwargs** – smoothing key words
- **di_kwargs** – dilation key words

Return mask the mask covering the identified RFI

seek.mitigation.sum_threshold.**get_sm_kwargs**(*kernel_m=40*, *kernel_n=20*, *sigma_m=7.5*, *sigma_n=15*)

Creates a dict with the smoothing keywords.

Parameters

- **kernel_m** – kernel window size in axis=1
- **kernel_n** – kernel window size in axis=0
- **sigma_m** – kernel sigma in axis=1
- **sigma_n** – kernel sigma in axis=0

Returns dictionary with the smoothing keywords

seek.mitigation.sum_threshold.**get_sumthreshold_kwargs**(*params*)
Creates the smoothing and dilation kwargs from a params objects.

Parameters **params** – the params object containing the configuration

Returns smoothing and dilation kwargs

seek.mitigation.sum_threshold.**normalize**(*data*, *mask*)
Simple normalization of standing waves: subtracting the median over time for each frequency.

Parameters

- **data** – data
- **mask** – mask

Returns normalized data

seek.mitigation.sum_threshold.**rm_rfi**(*ctx*)
Call the main SumThreshold routine.

Parameters **ctx** – context

Returns SumThreshold RFI mask.

sum_threshold_utils Module

Created on Dec 21, 2015

author: jakeret

seek.mitigation.sum_threshold_utils.**get_stats**(*rfi*, *rfi_mask*)
Returns the stats needed to compute a ROC curve.

Parameters

- **rfi** – array containing the RFI pixels
- **rfi_mask** – boolean array that masks the RFI

Returns rl, ml, il count of rfi pixels, count of masked pixels, count of intersecting pixels

seek.mitigation.sum_threshold_utils.**plot_data**(*data*, *ax*, *title*, *vmin=None*, *vmax=None*, *cb=True*, *norm=None*, *extent=None*, *cmap=None*)
Plot TOD.

seek.mitigation.sum_threshold_utils.**plot_dilation**(*st_mask*, *mask*, *dilated_mask*)
Plot mask and dilation.

seek.mitigation.sum_threshold_utils.**plot_moments**(*data*)
Plot standard divation and mean of data.

seek.mitigation.sum_threshold_utils.**plot_steps**(*data*, *st_mask*, *smoothed_data*, *res*, *eta*)
Plot individual steps of SumThreshold.

plugins Package

background_removal Module

calibration Module

Created on Feb 4, 2016

author: cchang

class seek.plugins.calibration.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

This class is used to specify which calibration type to use. If the case "data" is specified, derive gain curve from separate module in calibration directory.

create_maps Module

Created on Feb 26, 2016

author: jakeret

class seek.plugins.create_maps.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

This class fills the map pixels by delegating the computation to the 'map_maker'.

class seek.plugins.create_maps.**RestructuredTODStore**(*paths*)
Bases: object

This class restructures all the 'chunks' of data so that all the data points associated with the same healpix pixel is collected together.

get(*idx*)

find_nested_files Module

Created on Jul 27, 2015

author: jakeret

class seek.plugins.find_nested_files.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Traverses the file system from the *file_prefix* and collects all data and coord paths within the scanning strategy start and end date

seek.plugins.find_nested_files.**get_calibration_path**(*path*, *date*)
Get path for calibration file.

Parameters

- **path** – path for the data files
- **date** – date corresponding to the files
- **prefix** – file prefix

Returns full path to calibration file

seek.plugins.find_nested_files.**get_coords_path**(*path*, *date*, *prefix*)
Get path for coordinate file.

Parameters

- **path** – path for the data files

- **date** – date corresponding to the files
- **prefix** – file prefix

Returns full path to coordinate file

seek.plugins.find_nested_files.**is_calibration_day**(*path*, *date*)
Determine whether the date is a calibration day.

Parameters

- **path** – path where files are stored
- **date** – date corresponding to the files
- **prefix** – file prefix

Returns TRUE or FALSE

seek.plugins.find_nested_files.**is_skiped**(*path*, *date*, *prefix*)
Determine whether the date is skipped.

Parameters

- **path** – path where files are stored
- **date** – date corresponding to the files
- **prefix** – file prefix

Returns TRUE or FALSE

initialize Module

load_data Module

Created on Jul 28, 2015

author: jakeret

class seek.plugins.load_data.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Loads the data from files, applies cuts in frequency direction and also integrates the data in time and freq

class seek.plugins.load_data.**TimeOrderedData**(*strategy_start*, *frequencies*, *time_axis*, *vx*, *vy*, *ref_channel*)
Bases: tuple

frequencies
Alias for field number 1

ref_channel
Alias for field number 5

strategy_start
Alias for field number 0

time_axis
Alias for field number 2

vx
Alias for field number 3

vy
Alias for field number 4

seek.plugins.load_data.**convert_callisto**(*data*)
Converts the digits into kelvins

Parameters

- **frequencies** – the frequencies of the data
- **data** – array containing the data [freq, time]

Returns data the converted data

seek.plugins.load_data.**convert_to_radio_frequency**(*frequencies*, *spectrometer*)
Conversion between internal frequency and the actual physical frequency

Parameters

- **IF** – internal frequency array
- **ctx** – context object

Returns RF converted frequency array

seek.plugins.load_data.**get_observation_start**(*path*, *file_type*)
Extracts the observation date

Parameters **path** – path to the file

Returns observation_start datetime object with the date

seek.plugins.load_data.**get_observation_start_from_fits**(*path*)
Extracts the observation date

Parameters **path** – path to the file

Returns observation_start datetime object with the date

seek.plugins.load_data.**get_observation_start_from_hdf5**(*path*)
Extracts the observation date

Parameters **path** – path to the file

Returns observation_start datetime object with the date

seek.plugins.load_data.**load_tod**(*file_paths*, *ctx*)
Load the time ordered data from the given file paths :param file_paths: list of absolute file paths pointing to the data files :param ctx: Context object with the configuration

Returns TimeOrderedData returns a TimeOrderedData namedtupel

load_preprocessed_data Module

Created on Jan 15, 2016

author: jakeret

class seek.plugins.load_preprocessed_data.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Loads the data, mask and frequencies of the current iteration from disk. Can be used for closer analysis of the masking (sum threshold). The data is read from the current folder using the same filename as the first input filename.

make_maps Module

Created on Jan 5, 2015

author: seehars

class seek.plugins.make_maps.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Make map from restructured TOD based on the specified map_maker.

map_file_paths Module

Created on Feb 3, 2015

author: jakeret

class seek.plugins.map_file_paths.**Plugin**(*ctx*)
Bases: object

Maps the file paths to the plugin collection.

getWorkload()

seek.plugins.map_file_paths.**chunk**(*iterable*, *n*)

map_indicies Module

Created on Mar 7, 2016

author: jakeret

class seek.plugins.map_indicies.**Plugin**(*ctx*)
Bases: object

Maps the file paths to the plugin collection.

getWorkload()

seek.plugins.map_indicies.**chunk**(*iterable*, *n*)

mask_artefacts Module

Created on Jun 6, 2016

author: jakeret

class seek.plugins.mask_artefacts.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

First masks given frequency ranges that are known to be RFI-contaminatedm. Next masks artefacts based on a custom file in the data directory.

mask_artefacts()
Mask artefacts.

Returns mask after specified artefacts are masked.

mask_frequencies()
Mask bad frequency channels.

Returns mask after specified frequencies are masked.

mask_objects Module

Created on Feb 6, 2015

author: jakeret

class seek.plugins.mask_objects.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Masks the Sun and the Moon using ephem.

mask_objects()

seek.plugins.mask_objects.**get_object_separation**(*obs*, *start_date*, *time*, *ra*, *dec*)
Get separation between the Sun/Moon and the RA/DEC of a pixel.

Parameters

- **obs** – pyephem observer
- **start_date** – date
- **time** – time axis of TOD
- **ra** – RA for TOD
- **dec** – DEC for TOD

Returns separation of the TOD positions with the Sun and the Moon

post_process_tod Module

Created on Feb 6, 2015

author: jakeret

class seek.plugins.post_process_tod.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Writes the data, mask and frequencies of the current iteration to disk. Can be used for closer analysis of the masking (sum threshold). Output is written to the current folder using the same filename as the first input filename (may overwrite the original file if not being careful)

pre_process_tod Module

Created on Jan 20, 2016

author: jakeret

class seek.plugins.pre_process_tod.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Converts the TOD's depending on the spectrometer

seek.plugins.pre_process_tod.**apply_gain**(*frequencies*, *data*, *gain_file*)
Converts the digits into kelvins using a gain template

Parameters

- **frequencies** – the frequencies of the data
- **data** – array containing the data [freq, time]

Returns data the converted data

process_coords Module

Created on Jul 28, 2015

author: jakeret

class seek.plugins.process_coords.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Loads the telescope coordinate file for the current observation date and converts AZ/EL to RA/DEC.

seek.plugins.process_coords.**convert_coords**(*date*, *time_steps*, *azs*, *els*, *obs*)
Convert the time/az/ele coordinates to RA/DEC.

Parameters

- **date** – date
- **time_steps** – time interval between two TOD pixels
- **azs** – Azimuth coordinate
- **els** – Elevation coordinate
- **obs** – pyephem observer

Returns RA/DEC array corresponding to TOD

reduce_map_indicies Module

Created on Feb 26, 2016

author: jakeret

class seek.plugins.reduce_map_indicies.**Plugin**(*ctx*)
Bases: object

Reduces the restructured TODs constructed per chunk to one.

reduce(*ctxList*)

reduce_maps Module

remove_RFI Module

Created on Jan 5, 2015

author: seehars

class seek.plugins.remove_RFI.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Call the specified RFI mitigation module.

restructure_tod Module

write_maps Module

Created on Jan 5, 2015

author: seehars

class seek.plugins.write_maps.**Plugin**(*ctx*, ***kwargs*)
Bases: ivy.plugin.base_plugin.BasePlugin

Writes map and associated information to HDF5 files.

utils Package

utils Package

seek.utils.**format_date**(*date*, *fmt='%Y-%m-%d'*)
Format datetime object into specific format string.

Parameters

- **date** – datetime object
- **fmt** – format definition

Returns formatted string

seek.utils.**load_file**(*path*, ***kwargs*)
Load text file within the main code directory.

Parameters **path** – path of file within the main code directory

Returns array generated from text file

seek.utils.**parse_datetime**(*s*, *fmt='%Y-%m-%d-%H:%M:%S'*)
Parse datetime object with specific format.

Parameters

- **s** – datetime string
- **fmt** – format definition

Returns parsed datetime object

filter Module

Created on Jan 23, 2015

author: jakeret

seek.utils.filter.**gaussian_filter**(*V*, *mask*, *M=40*, *N=20*, *sigma_m=0.5*, *sigma_n=0.5*)
Applies a gaussian filter (smoothing) to the given array taking into account masked values :param V: the value array to be smoothed :param mask: boolean array defining masked values :param M: kernel window size in axis=1 :param N: kernel window size in axis=0 :param sigma_m: kernel sigma in axis=1 :param sigma_n: kernel sigma in axis=0

Returns vs the filtered array

plotting Module

sphere Module

Created on Dec 22, 2014

author: jakeret

seek.utils.sphere.**altaz_to_ra_dec**(*date*, *az*, *alt*, *obs=None*, *params=None*)
Convert Azimuth/Elevation/time to RA/DEC coordinates.

Parameters

- **date** – date and time
- **az** – Azimuth coordinate
- **alt** – Elevation coordinate
- **obs** – pyephem observer
- **params** – context parameters

Returns RA/DEC coordinates

seek.utils.sphere.**get_observer**(*params*)
Define pyephem observer.

Parameters **params** – telescope parameters from context

Returns pyephem observer

tod_utils Module

Created on Aug 18, 2015

author: jakeret

seek.utils.tod_utils.**get_empty_mask**(*shape*)

seek.utils.tod_utils.**smooth**(*tod*, *factor*, *axis=1*)
Smoothes a tod by the given factor. E.g. (axis=1):

```
[[1,1,2,2,3,3], -- factor 2 --> [[1,2,3],
[4,4,5,5,6,6]]                  [4,5,6]]
```

Parameters

- **tod** – the data to be smoothed
- **factor** – the factor to use for the integration

seek.utils.tod_utils.**spectral_kurtosis_mask**(*p_phase0*, *p_phase1*, *p2_phase0*, *p2_phase1*, *M*, *offset*)
Creates a mask using the spectral kurtosis for the given arrays

Parameters

- **p_phase0** – array of P values for phase0
- **p_phase1** – array of P values for phase1
- **p2_phase0** – array of P^2 values for phase0

- **p2_phase1** – array of P^2 values for phase1
- **M** – number of accumulations
- **offset** – P_select 0

Returns mask boolean array containing the mask

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

Types of Contributions

Report Bugs

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Implement Features

Write Documentation

SEEK: Signal Extraction and Emission Kartographer could always use more documentation, whether as part of the official SEEK: Signal Extraction and Emission Kartographer docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.

3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

```
$ py.test test/test_seek.py
```

Credits

Development Lead

- Joel Akeret <jakeret@phys.ethz.ch>
- Sebastian Seehars <seehars@phys.ethz.ch>
- Chihway Chang <chihway.chang@phys.ethz.ch>

Contributors

None yet. Why not be the first?

History

0.1.0 (2016-07-20)

- Publication

CHAPTER 2

Feedback

If you have any suggestions or questions about **SEEK: Signal Extraction and Emission Kartographer** feel free to email me at jakeret@phys.ethz.ch.

If you encounter any errors or problems with **SEEK: Signal Extraction and Emission Kartographer**, please let me know!

Python Module Index

s

seek.__init__, 5
seek.calibration.astro_calibration_source, 5
seek.calibration.fitting, 6
seek.calibration.flux_calibration_transit, 6
seek.config.common, 8
seek.config.process_survey, 9
seek.config.process_survey_fft, 9
seek.mapmaking.filter_mapper, 9
seek.mapmaking.filter_variance_mapper, 9
seek.mapmaking.simple_mapper, 10
seek.mapmaking.simple_variance_mapper, 10
seek.mitigation.outlier_masking, 10
seek.mitigation.sum_threshold, 10
seek.mitigation.sum_threshold_utils, 12
seek.plugins.calibration, 13
seek.plugins.create_maps, 13
seek.plugins.find_nested_files, 13
seek.plugins.load_data, 14
seek.plugins.load_preprocessed_data, 15
seek.plugins.make_maps, 16
seek.plugins.map_file_paths, 16
seek.plugins.map_indicies, 16
seek.plugins.mask_artefacts, 16
seek.plugins.mask_objects, 17
seek.plugins.post_process_tod, 17
seek.plugins.pre_process_tod, 17
seek.plugins.process_coords, 18
seek.plugins.reduce_map_indicies, 18
seek.plugins.remove_RFI, 18
seek.plugins.write_maps, 19
seek.utils, 19
seek.utils.filter, 19
seek.utils.sphere, 20
seek.utils.tod_utils, 20

Index

A

altaz_to_ra_dec() (in module seek.utils.sphere), 20
apply_gain() (in module seek.plugins.pre_process_tod), 17
azimut (seek.calibration.flux_calibration_transit.CalibrationSource attribute), 6

B

barrs77_power_law() (in module seek.calibration.astro_calibration_source), 5
benz_sun() (in module seek.calibration.astro_calibration_source), 5
binary_mask_dilation() (in module seek.mitigation.sum_threshold), 10
black_body() (in module seek.calibration.astro_calibration_source), 5

C

calibrate() (in module seek.calibration.flux_calibration_transit), 7
CalibrationSource (class in seek.calibration.flux_calibration_transit), 6
chunk() (in module seek.plugins.map_file_paths), 16
chunk() (in module seek.plugins.map_indicies), 16
convert_callisto() (in module seek.plugins.load_data), 15
convert_coords() (in module seek.plugins.process_coords), 18
convert_to_radio_frequency() (in module seek.plugins.load_data), 15
Coords (class in seek.__init__), 5

D

date (seek.calibration.flux_calibration_transit.CalibrationSource attribute), 6

E

elevation (seek.calibration.flux_calibration_transit.CalibrationSource attribute), 6

F

filter_data() (in module seek.mapmaking.filter_mapper), 9
fit_func() (in module seek.calibration.fitting), 6
fit_gaussian_source() (in module seek.calibration.flux_calibration_transit), 7
fit_gaussian_source_full() (in module seek.calibration.flux_calibration_transit), 7
fit_sun_to_gain() (in module seek.calibration.flux_calibration_transit), 8
format_date() (in module seek.utils), 19
frequencies (seek.plugins.load_data.TimeOrderedData attribute), 14

G

gauss2() (in module seek.calibration.fitting), 6
gauss_A (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
gauss_A_err (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
gauss_sigma (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
gauss_sigma_err (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
gauss_x0 (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
gauss_x0_err (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 6
GaussFitResult (class in seek.calibration.flux_calibration_transit), 6
gaussian_filter() (in module seek.utils.filter), 19

get() (seek.plugins.create_maps.RestructuredTODStore method), 13
get_calibration_path() (in module seek.plugins.find_nested_files), 13
get_coords_path() (in module seek.plugins.find_nested_files), 13
get_di_kwrags() (in module seek.mitigation.sum_threshold), 11
get_empty_mask() (in module seek.utils.tod_utils), 20
get_mapped_values() (in module seek.mapmaking.filter_mapper), 9
get_mapped_values() (in module seek.mapmaking.filter_variance_mapper), 9
get_mapped_values() (in module seek.mapmaking.simple_mapper), 10
get_mapped_values() (in module seek.mapmaking.simple_variance_mapper), 10
get_object_separation() (in module seek.plugins.mask_objects), 17
get_observation_start() (in module seek.plugins.load_data), 15
get_observation_start_from_fits() (in module seek.plugins.load_data), 15
get_observation_start_from_hdf5() (in module seek.plugins.load_data), 15
get_observer() (in module seek.utils.sphere), 20
get_rfi_mask() (in module seek.mitigation.sum_threshold), 11
get_sm_kwargs() (in module seek.mitigation.sum_threshold), 11
get_stats() (in module seek.mitigation.sum_threshold_utils), 12
get_sumthreshold_kwargs() (in module seek.mitigation.sum_threshold), 11
getMask() (in module seek.mitigation.outlier_masking), 10
getWorkload() (seek.plugins.map_file_paths.Plugin method), 16
getWorkload() (seek.plugins.map_indicies.Plugin method), 16

I

is_calibration_day() (in module seek.plugins.find_nested_files), 14
is_skiped() (in module seek.plugins.find_nested_files), 14

L

load_file() (in module seek.utils), 19
load_tod() (in module seek.plugins.load_data), 15

M

mask_artefacts() (seek.plugins.mask_artefacts.Plugin method), 16
mask_frequencies() (seek.plugins.mask_artefacts.Plugin method), 16
mask_objects() (seek.plugins.mask_objects.Plugin method), 17

N

normalize() (in module seek.mitigation.sum_threshold), 12

P

parse_datetime() (in module seek.utils), 19
plot_data() (in module seek.mitigation.sum_threshold_utils), 12
plot_dilation() (in module seek.mitigation.sum_threshold_utils), 12
plot_moments() (in module seek.mitigation.sum_threshold_utils), 12
plot_steps() (in module seek.mitigation.sum_threshold_utils), 12
Plugin (class in seek.plugins.calibration), 13
Plugin (class in seek.plugins.create_maps), 13
Plugin (class in seek.plugins.find_nested_files), 13
Plugin (class in seek.plugins.load_data), 14
Plugin (class in seek.plugins.load_preprocessed_data), 15
Plugin (class in seek.plugins.make_maps), 16
Plugin (class in seek.plugins.map_file_paths), 16
Plugin (class in seek.plugins.map_indicies), 16
Plugin (class in seek.plugins.mask_artefacts), 16
Plugin (class in seek.plugins.mask_objects), 17
Plugin (class in seek.plugins.post_process_tod), 17
Plugin (class in seek.plugins.pre_process_tod), 17
Plugin (class in seek.plugins.process_coords), 18
Plugin (class in seek.plugins.reduce_map_indicies), 18
Plugin (class in seek.plugins.remove_RFI), 18
Plugin (class in seek.plugins.write_maps), 19
process_calibration_transits() (in module seek.calibration.flux_calibration_transit), 8
process_source() (in module seek.calibration.flux_calibration_transit), 8

R

reduce() (seek.plugins.reduce_map_indicies.Plugin method), 18
ref_channel (seek.plugins.load_data.TimeOrderedData attribute), 14
RestructuredTODStore (class in seek.plugins.create_maps), 13
rm_rfi() (in module seek.mitigation.outlier_masking), 10
rm_rfi() (in module seek.mitigation.sum_threshold), 12
rsquared (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 7

S

seek.__init__ (module), 5
seek.calibration.astro_calibration_source (module), 5
seek.calibration.fitting (module), 6
seek.calibration.flux_calibration_transit (module), 6
seek.config.common (module), 8
seek.config.process_survey (module), 9
seek.config.process_survey_fft (module), 9
seek.mapmaking.filter_mapper (module), 9
seek.mapmaking.filter_variance_mapper (module), 9
seek.mapmaking.simple_mapper (module), 10
seek.mapmaking.simple_variance_mapper (module), 10
seek.mitigation.outlier_masking (module), 10
seek.mitigation.sum_threshold (module), 10
seek.mitigation.sum_threshold_utils (module), 12
seek.plugins.calibration (module), 13
seek.plugins.create_maps (module), 13
seek.plugins.find_nested_files (module), 13
seek.plugins.load_data (module), 14
seek.plugins.load_preprocessed_data (module), 15
seek.plugins.make_maps (module), 16
seek.plugins.map_file_paths (module), 16
seek.plugins.map_indicies (module), 16
seek.plugins.mask_artefacts (module), 16
seek.plugins.mask_objects (module), 17
seek.plugins.post_process_tod (module), 17
seek.plugins.pre_process_tod (module), 17
seek.plugins.process_coords (module), 18
seek.plugins.reduce_map_indicies (module), 18
seek.plugins.remove_RFI (module), 18
seek.plugins.write_maps (module), 19
seek.utils (module), 19
seek.utils.filter (module), 19
seek.utils.sphere (module), 20
seek.utils.tod_utils (module), 20
sky1 (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 7
sky1_err (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 7
sky2 (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 7
sky2_err (seek.calibration.flux_calibration_transit.GaussFitResult attribute), 7
smooth() (in module seek.utils.tod_utils), 20
source() (in module seek.calibration.astro_calibration_source), 5
spectral_kurtosis_mask() (in module seek.utils.tod_utils), 20
strategy_start (seek.plugins.load_data.TimeOrderedData attribute), 14

T

target (seek.calibration.flux_calibration_transit.CalibrationSource attribute), 6
time_axis (seek.plugins.load_data.TimeOrderedData attribute), 14
TimeOrderedData (class in seek.plugins.load_data), 14

V

vx (seek.plugins.load_data.TimeOrderedData attribute), 14
vy (seek.plugins.load_data.TimeOrderedData attribute), 14