PyInSAR

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 pyinsar Namespace Reference

Namespaces

- data_import
- output
- processing

5.2 pyinsar.data_import Namespace Reference

Namespaces

- import_georaster
- import_raster
- import_srcmod
- import_utils
- sentinel
- uavsar

5.3 pyinsar.data_import.import_georaster Namespace Reference

Functions

- def open_georaster (georaster_path, read_only=True)

  Open a georaster with GDAL.
- def get_georaster_array (gdal_georaster, remove_ndv=True, as_float=True)

  Get a NumPy array from a georaster opened with GDAL.
- def get_georaster_extent (gdal_georaster)

  Get the extent of a georaster opened with GDAL.
- def print_georaster_info (gdal_georaster)

  Print some information about the GDAL georaster.

5.3.1 Function Documentation

5.3.1.1 get_georaster_array()

```
def pyinsar.data_import.import_georaster.get_georaster_array (
              gdal_georaster,
              remove_ndv = True,
              as_float = True )
```

Get a NumPy array from a georaster opened with GDAL.

Parameters

gdal_georaster	A georaster opened with GDAL
remove_ndv	Replace the no-data value as mentionned in the label by np.nan
as_float	Transform the array to a float array

Returns

The array

5.3.1.2 get_georaster_extent()

```
def pyinsar.data_import.import_georaster.get_georaster_extent (
              gdal_georaster )
```

Get the extent of a georaster opened with GDAL.

Parameters

gdal_georaster	A georaster opened with GDAL

Returns

The georaster extent

5.3.1.3 open_georaster()

```
def pyinsar.data_import.import_georaster.open_georaster (
              georaster_path,
              read_only = True )
```

Open a georaster with GDAL.

Parameters

georaster_path	Location of the georaster
read_only	Determine if the georaster can be modified

Returns

The georaster as a GDAL data set

5.3.1.4 print_georaster_info()

```
def pyinsar.data_import.import_georaster.print_georaster_info (
              gdal_georaster )
```

Print some information about the GDAL georaster.

Parameters

gdal_georaster	A georaster opened with GDAL

5.4 pyinsar.data_import.import_raster Namespace Reference

Functions

- def read_rsc_header_file (file_path)

  Import GACOS runs.
- def open_gacos_tropospheric_delays (tropodelay_header_path)

  Open a topospheric delay map computed by the Generic Atmospheric Correction Online Service for InSAR (GACOS)
- def open_sgems_file (file_location)

  Import SGEMS files.
- def open_sgems_file_from_url (file_url)

  Open an SGEMS file containing one or several variables in an array from the file's URL.

5.4.1 Function Documentation

5.4.1.1 open_gacos_tropospheric_delays()

```
def pyinsar.data_import.import_raster.open_gacos_tropospheric_delays (
              tropodelay_header_path )
```

Open a topospheric delay map computed by the Generic Atmospheric Correction Online Service for InSAR (GACOS)

Parameters

tropodelay_header_path	Path to the header file (.ztd.rsc or .dztd.rsc)

Returns

A NumPy array containing the topospheric delay in meters and a tuple containing the extent of the array

5.4.1.2 open_sgems_file()

```
def pyinsar.data_import.import_raster.open_sgems_file (
              file_location )
```

Import SGEMS files.

Open an SGEMS file containing one or several variables in an array

```
@param file_location: The location of the file

@return A NumPy array
```

5.4.1.3 open_sgems_file_from_url()

```
def pyinsar.data_import.import_raster.open_sgems_file_from_url (
              file_url )
```

Open an SGEMS file containing one or several variables in an array from the file's URL.

Parameters

file_url	The URL of the file

Returns

A NumPy array

5.4.1.4 read_rsc_header_file()

```
def pyinsar.data_import.import_raster.read_rsc_header_file (
              file_path )
```

Import GACOS runs.

Read the rsc header file from GACOS data

```
@param file_location: The path to the file

@return A dictionary containing the header's information
```

5.5 pyinsar.data_import.import_srcmod Namespace Reference

Functions

- def read_srcmod_data (srcmod_data, dtype=np.float64, skip_sanity_check=False)

    ∗∗∗ In Development ∗∗∗ Generate faults of okada sources from src mod mat files.

5.5.1 Function Documentation

5.5.1.1 read_srcmod_data()

```
def pyinsar.data_import.import_srcmod.read_srcmod_data (
              srcmod_data,
              dtype = np.float64,
              skip_sanity_check = False )
```

∗∗∗ In Development ∗∗∗ Generate faults of okada sources from src mod mat files.

Note

Only single segment models with a single time window are currently supported

Parameters

srcmod_data	src mod data read in from the .mat file
dtype	Data type to use
skip_sanity_check	Skip checks to ensure data was interpreted properly (Used for debugging)

Returns

List of faults objects, list of slips, list of rakes

5.6 pyinsar.data_import.import_utils Namespace Reference

Functions

- def download_file (url, folder_path, username=None, password=None, filename=None)
  Download a file from a URL.

5.6.1 Function Documentation

5.6.1.1 download_file()

```
def pyinsar.data_import.import_utils.download_file (
              url,
              folder_path,
              username = None,
              password = None,
              filename = None )
```

Download a file from a URL.

Parameters

url	The URL where the file is
folder_path	Path to the folder where the downloaded file will be stored
username	username for authentification, if needed
password	Password for authentification, if needed
filename	Change the filename, if needed

Returns

The file path if download was succesful, none otherwise

5.7 pyinsar.data_import.sentinel Namespace Reference

Functions

- def parse_satellite_data (in_satellite_file)

  Parse Sentinel satellite data.
- def get_url_precise_orbit (product_name)
- def download_precise_orbits (product_folder, orbit_folder, username, password)

  Download the precise orbits for all the Sentinel-1 products in a folder.
- def download_products (product_names, product_folder, base_url='https://datapool.asf.alaska.edu/SLC', use_↩ vertex=True, username=None, password=None)

  Download Sentinel-1 products in a folder.

5.7.1 Function Documentation

5.7.1.1 download_precise_orbits()

```
def pyinsar.data_import.sentinel.download_precise_orbits (
              product_folder,
              orbit_folder,
              username,
              password )
```

Download the precise orbits for all the Sentinel-1 products in a folder.

Parameters

product_folder	The folder where the Sentinel-1 products are
orbit_folder	The folder where to put the orbit files
username	The username for authentification on Earthdata
password	The password for authentification on Earthdata

Returns

The paths of the orbit files, none if a file couldnot be downloaded

5.7.1.2 download_products()

```
def pyinsar.data_import.sentinel.download_products (
              product_names,
              product_folder,
              base_url = 'https://datapool.asf.alaska.edu/SLC',
              use_vertex = True,
              username = None,
              password = None )
```

Download Sentinel-1 products in a folder.

Parameters

product_names	List of Sentinel-1 product names
product_folder	The folder where to put the product files
base_url	Base url from where to download the files (default is from the Alaska Satellite Facility)
use_vertex	True if the base url is that of the Alaska Satellite Facility
username	The username for authentification on Earthdata
password	The password for authentification on Earthdata

Returns

The paths of the orbit files, none if a file couldnot be downloaded

5.7.1.3 get_url_precise_orbit()

```
def pyinsar.data_import.sentinel.get_url_precise_orbit (
              product_name )
```

5.7.1.4 parse_satellite_data()

```
def pyinsar.data_import.sentinel.parse_satellite_data (
              in_satellite_file )
```

Parse Sentinel satellite data.

Parameters

in_satellite_file	Satellite orbit filename

Returns

DataFrame of orbit information

5.8 pyinsar.data_import.uavsar Namespace Reference

Functions

- def read_uavsar_metadata (in_file)

  Parse UAVSAR metadata.

5.8.1 Function Documentation

5.8.1.1 read_uavsar_metadata()

```
def pyinsar.data_import.uavsar.read_uavsar_metadata (
              in_file )
```

Parse UAVSAR metadata.

Parameters

in_file	String of Metadata filename or file object (file should end in .ann)

Returns

OrderedDict of metadata

5.9 pyinsar.output Namespace Reference

Namespaces

- export_georaster
- plot_raster

5.10 pyinsar.output.export_georaster Namespace Reference

Functions

- def create_georaster_from_array (georaster_array, geotransform, projection, file_type='MEM', file_path='', data↩_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0., options=[ ])

  Create a GDAL georaster from a Numpy array.

5.10.1 Function Documentation

5.10.1.1 create_georaster_from_array()

```
def pyinsar.output.export_georaster.create_georaster_from_array (
              georaster_array,
              geotransform,
              projection,
              file_type = 'MEM',
              file_path = '',
              data_type = gdal.GDT_Float64,
              no_data_value = -99999.,
              scale = 1.,
              offset = 0.,
              options = [] )
```

Create a GDAL georaster from a Numpy array.

Parameters

georaster_array	The Numpy array
geotransform	The extent and cell spacing of the georaster
projection	The projection of the georaster
file_type	Type to save the file (default is memory)
file_path	Where to store the new georaster (default is memory)
data_type	Data type of the georaster
no_data_value	No data value for the georaster
scale	Scaling factor for the georaster
offset	Offset factor for the georaster
options	List of options for compression

Returns

The GDAL georaster

5.11 pyinsar.output.plot_raster Namespace Reference

Functions

- def average_minmax_slices (array, axis=0)
- def plot_interactive_slicing (array, slice_index, model_array=None, axis=0, cmap='viridis', extent=None, clabel='', xlabel='', ylabel='', figsize=None, update_colorbar=False)
- def plot_interactive_multiple_slicing (array, axes, slice_indexes, model_array=None, cmap='viridis', update_colorbar=False, vmin=0., vmax=1., extent=None, clabel='', xlabel='', ylabel='', figsize=None)

5.11.1 Function Documentation

5.11.1.1 average_minmax_slices()

```
def pyinsar.output.plot_raster.average_minmax_slices (
              array,
              axis = 0 )
```

5.11.1.2 plot_interactive_multiple_slicing()

```
def pyinsar.output.plot_raster.plot_interactive_multiple_slicing (
              array,
              axes,
              slice_indexes,
              model_array = None,
              cmap = 'viridis',
              update_colorbar = False,
              vmin = 0.,
              vmax = 1.,
              extent = None,
              clabel = '',
              xlabel = '',
              ylabel = '',
              figsize = None )
```

5.11.1.3 plot_interactive_slicing()

```
def pyinsar.output.plot_raster.plot_interactive_slicing (
              array,
              slice_index,
              model_array = None,
              axis = 0,
              cmap = 'viridis',
              extent = None,
              clabel = '',
              xlabel = '',
              ylabel = '',
              figsize = None,
              update_colorbar = False )
```

5.12 pyinsar.processing Namespace Reference

Namespaces

- corrections
- data_fetcher
- deformation
- discovery
- geography
- instruments
- isce
- machine_learning
- utilities

5.13 pyinsar.processing.corrections Namespace Reference

Namespaces

- topography
- troposphere

5.14 pyinsar.processing.corrections.topography Namespace Reference

Functions

- def ellipsoidal_earth_slant_ranges (azimuth_time, latlon, orbit_interp, start_x, end_x, start_y, end_y)

  Compute slant ranges assuming no topography.

5.14.1 Function Documentation

5.14.1.1 ellipsoidal_earth_slant_ranges()

```
def pyinsar.processing.corrections.topography.ellipsoidal_earth_slant_ranges (
              azimuth_time,
              latlon,
              orbit_interp,
              start_x,
              end_x,
              start_y,
              end_y )
```

Compute slant ranges assuming no topography.

Parameters

azimuth_time	Pandas time series data conatining the time of each azimuth line
latlon	Function to compute latitude and longitude for each pixel coordinate
orbit_interp	Function to compute satellite positions
start_x	Starting x pixel
end_x	Ending pixel x pxiel
start_y	Starting y pixel
end_y	Endying y pixel

Returns

Slant range distance to each pixel

5.15 pyinsar.processing.corrections.troposphere Namespace Reference

Functions

- def vapor_pressure (T)

  Under development.
- def N (P, T, RH, k1=77.6, k2=23.3, k3=3.75E5)

  Under development.
- def N_h (h, P, T, RH, k1=77.6, k2=23.3, k3=3.75E5)

  Under development.
- def compute_delays (h, P, T, RH)

  Under development.

5.15.1 Function Documentation

5.15.1.1 compute_delays()

```
def pyinsar.processing.corrections.troposphere.compute_delays (
              h,
              P,
              T,
              RH )
```

Under development.

5.15.1.2 N()

```
def pyinsar.processing.corrections.troposphere.N (
              P,
              T,
              RH,
              k1 = 77.6,
              k2 = 23.3,
              k3 = 3.75E5 )
```

Under development.

5.15.1.3 N_h()

```
def pyinsar.processing.corrections.troposphere.N_h (
              h,
              P,
              T,
              RH,
              k1 = 77.6,
              k2 = 23.3,
              k3 = 3.75E5 )
```

Under development.

5.15.1.4 vapor_pressure()

```
def pyinsar.processing.corrections.troposphere.vapor_pressure (
              T )
```

Under development.

5.16 pyinsar.processing.data_fetcher Namespace Reference

Namespaces

- gdal
- hdf_retriever
- okada

5.17 pyinsar.processing.data_fetcher.gdal Namespace Reference

Classes

- class GDAL_DataFetcher

  Data fetcher for loading Images produced compatiable with GDAL.

5.18 pyinsar.processing.data_fetcher.hdf_retriever Namespace Reference

Classes

- class DataRetriever

  Data fetcher for retrieving hdf image data made for training in convolutional neural networks.

5.19 pyinsar.processing.data_fetcher.okada Namespace Reference

Classes

- class DataFetcher

  Generates data from an Okada model.

5.20 pyinsar.processing.deformation Namespace Reference

Namespaces

- elastic_halfspace

5.21 pyinsar.processing.deformation.elastic_halfspace Namespace Reference

Namespaces

- fault
- mogi
- okada
- pipe
- surface_load

5.22 pyinsar.processing.deformation.elastic_halfspace.fault Namespace Reference

Classes

- class Fault

  ∗∗∗ In Development ∗∗∗ Model a fault as a collection of small okada faults

5.23 pyinsar.processing.deformation.elastic_halfspace.mogi Namespace Reference

Functions

- def compute_mogi_source_displacement (source_x, source_y, source_depth, source_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

5.23.1 Function Documentation

5.23.1.1 compute_mogi_source_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.mogi.compute_mogi_source_displacement (
              source_x,
              source_y,
              source_depth,
              source_radius,
              poisson_ratio,
              pressurization,
              shear_modulus,
              xx_array,
              yy_array )
```

5.24 pyinsar.processing.deformation.elastic_halfspace.okada Namespace Reference

Functions

- def I1 (xi, eta, q, delta, nu, R, X, d_tild)

  Okada's surface displacement.
- def I2 (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def I3 (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def I4 (xi, eta, q, delta, nu, R, d_tild)
- def I5 (xi, eta, q, delta, nu, R, X, d_tild)
- def f_x_strike (xi, eta, q, delta, nu)
- def f_x_dip (xi, eta, q, delta, nu)

- def f_x_tensile (xi, eta, q, delta, nu)
- def f_y_strike (xi, eta, q, delta, nu)
- def f_y_dip (xi, eta, q, delta, nu)
- def f_y_tensile (xi, eta, q, delta, nu)
- def f_z_strike (xi, eta, q, delta, nu)
- def f_z_dip (xi, eta, q, delta, nu)
- def f_z_tensile (xi, eta, q, delta, nu)
- def chinnerys_notation (f, x, p, q, L, W, delta, nu)
- def compute_okada_displacement (fault_centroid_x, fault_centroid_y, fault_centroid_depth, fault_strike, fault_dip, fault_length, fault_width, fault_rake, fault_slip, fault_open, poisson_ratio, xx_array, yy_array)
- def I1_int (xi, eta, z, y, delta, c, d, q, R)

  Okada's internal displacement.
- def I2_int (xi, eta, z, y, delta, c, d, q, R)
- def I3_int (xi, eta, z, y, delta, c, d, q, R)
- def I4_int (xi, eta, z, y, delta, c, d, q, R)
- def fA_1_strike (xi, eta, z, y, delta, c, alpha)
- def fA_2_strike (xi, eta, z, y, delta, c, alpha)
- def fA_3_strike (xi, eta, z, y, delta, c, alpha)
- def fB_1_strike (xi, eta, z, y, delta, c, alpha)
- def fB_2_strike (xi, eta, z, y, delta, c, alpha)
- def fB_3_strike (xi, eta, z, y, delta, c, alpha)
- def fC_1_strike (xi, eta, z, y, delta, c, alpha)
- def fC_2_strike (xi, eta, z, y, delta, c, alpha)
- def fC_3_strike (xi, eta, z, y, delta, c, alpha)
- def fA_1_dip (xi, eta, z, y, delta, c, alpha)
- def fA_2_dip (xi, eta, z, y, delta, c, alpha)
- def fA_3_dip (xi, eta, z, y, delta, c, alpha)
- def fB_1_dip (xi, eta, z, y, delta, c, alpha)
- def fB_2_dip (xi, eta, z, y, delta, c, alpha)
- def fB_3_dip (xi, eta, z, y, delta, c, alpha)
- def fC_1_dip (xi, eta, z, y, delta, c, alpha)
- def fC_2_dip (xi, eta, z, y, delta, c, alpha)
- def fC_3_dip (xi, eta, z, y, delta, c, alpha)
- def fA_1_tensile (xi, eta, z, y, delta, c, alpha)
- def fA_2_tensile (xi, eta, z, y, delta, c, alpha)
- def fA_3_tensile (xi, eta, z, y, delta, c, alpha)
- def fB_1_tensile (xi, eta, z, y, delta, c, alpha)
- def fB_2_tensile (xi, eta, z, y, delta, c, alpha)
- def fB_3_tensile (xi, eta, z, y, delta, c, alpha)
- def fC_1_tensile (xi, eta, z, y, delta, c, alpha)
- def fC_2_tensile (xi, eta, z, y, delta, c, alpha)
- def fC_3_tensile (xi, eta, z, y, delta, c, alpha)
- def fA_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fA_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fA_3 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fB_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fB_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fB_3 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fC_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fC_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def fC_3 (displacement_type, xi, eta, z, y, delta, c, alpha)

- def chinnerys_notation_int (f, displacement_type, x, y, z, L, W, delta, c, alpha)
- def compute_fault_internal_displacement_type (displacement_type, c, L, W, delta, U, alpha, xxx_array, yyy_array, zzz_array)
- def compute_okada_internal_displacement (fault_centroid_x, fault_centroid_y, fault_centroid_depth, fault_strike, fault_dip, fault_length, fault_width, fault_rake, fault_slip, fault_open, poisson_ratio, xxx_array, yyy_array, depth_↩ array)

5.24.1 Function Documentation

5.24.1.1 chinnerys_notation()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.chinnerys_notation (
              f,
              x,
              p,
              q,
              L,
              W,
              delta,
              nu )
```

5.24.1.2 chinnerys_notation_int()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.chinnerys_notation_int (
              f,
              displacement_type,
              x,
              y,
              z,
              L,
              W,
              delta,
              c,
              alpha )
```

5.24.1.3 compute_fault_internal_displacement_type()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.compute_fault_internal_displacement_↩
type (
              displacement_type,
              c,
              L,
              W,
              delta,
              U,
              alpha,
              xxx_array,
              yyy_array,
              zzz_array )
```

5.24.1.4 compute_okada_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.compute_okada_displacement (
              fault_centroid_x,
              fault_centroid_y,
              fault_centroid_depth,
              fault_strike,
              fault_dip,
              fault_length,
              fault_width,
              fault_rake,
              fault_slip,
              fault_open,
              poisson_ratio,
              xx_array,
              yy_array )
```

5.24.1.5 compute_okada_internal_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.compute_okada_internal_displacement (
              fault_centroid_x,
              fault_centroid_y,
              fault_centroid_depth,
              fault_strike,
              fault_dip,
              fault_length,
              fault_width,
              fault_rake,
              fault_slip,
              fault_open,
              poisson_ratio,
              xxx_array,
              yyy_array,
              depth_array )
```

5.24.1.6 f_x_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_dip (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.7 f_x_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_strike (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.8 f_x_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_tensile (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.9 f_y_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_dip (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.10 f_y_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_strike (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.11 f_y_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_tensile (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.12 f_z_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_dip (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.13 f_z_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_strike (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.14 f_z_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_tensile (
              xi,
              eta,
              q,
              delta,
              nu )
```

5.24.1.15 fA_1()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.16 fA_1_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.17 fA_1_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.18 fA_1_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.19 fA_2()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.20 fA_2_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.21 fA_2_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.22 fA_2_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.23 fA_3()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.24 fA_3_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.25 fA_3_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.26 fA_3_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.27 fB_1()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.28 fB_1_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.29 fB_1_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.30 fB_1_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.31 fB_2()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.32 fB_2_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.33 fB_2_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.34 fB_2_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.35 fB_3()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.36 fB_3_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.37 fB_3_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.38 fB_3_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.39 fC_1()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.40 fC_1_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.41 fC_1_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.42 fC_1_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.43 fC_2()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.44 fC_2_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.45 fC_2_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.46 fC_2_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.47 fC_3()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3 (
              displacement_type,
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.48 fC_3_dip()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_dip (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.49 fC_3_strike()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_strike (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.50 fC_3_tensile()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_tensile (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              alpha )
```

5.24.1.51 I1()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I1 (
              xi,
              eta,
              q,
              delta,
              nu,
              R,
              X,
              d_tild )
```

Okada's surface displacement.

5.24.1.52 I1_int()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I1_int (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              d,
              q,
              R )
```

Okada's internal displacement.

5.24.1.53 I2()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I2 (
              xi,
              eta,
              q,
              delta,
              nu,
              R,
              y_tild,
              d_tild )
```

5.24.1.54 I2_int()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I2_int (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              d,
              q,
              R )
```

5.24.1.55 I3()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I3 (
              xi,
              eta,
              q,
              delta,
              nu,
              R,
              y_tild,
              d_tild )
```

5.24.1.56 I3_int()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I3_int (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              d,
              q,
              R )
```

5.24.1.57 I4()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I4 (
              xi,
              eta,
              q,
              delta,
              nu,
              R,
              d_tild )
```

5.24.1.58 I4_int()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I4_int (
              xi,
              eta,
              z,
              y,
              delta,
              c,
              d,
              q,
              R )
```

5.24.1.59 I5()

```
def pyinsar.processing.deformation.elastic_halfspace.okada.I5 (
              xi,
              eta,
              q,
              delta,
              nu,
              R,
              X,
              d_tild )
```

5.25 pyinsar.processing.deformation.elastic_halfspace.pipe Namespace Reference

Functions

- def compute_closed_pipe_displacement (closed_pipe_x, closed_pipe_y, closed_pipe_depth_1, closed_pipe_↩ depth_2, closed_pipe_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for a closed pipe.
- def compute_open_pipe_displacement (open_pipe_x, open_pipe_y, open_pipe_depth_0, open_pipe_depth_1, open_pipe_depth_2, open_pipe_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for an open pipe.

5.25.1 Function Documentation

5.25.1.1 compute_closed_pipe_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.pipe.compute_closed_pipe_displacement (
              closed_pipe_x,
              closed_pipe_y,
              closed_pipe_depth_1,
              closed_pipe_depth_2,
              closed_pipe_radius,
              poisson_ratio,
              pressurization,
              shear_modulus,
              xx_array,
              yy_array )
```

Compute the surface displacements for a closed pipe.

Parameters

closed_pipe_x	x cooordinate for the pipe's center
closed_pipe_y	y cooordinate for the pipe's center
closed_pipe_depth↩ _1	Pipe's top depth
closed_pipe_depth↩ _2	Pipe's bottom depth
closed_pipe_radius	Pipe's radius
poisson_ratio	Poisson's ratio
pressurization	Change of pressure applied to the pipe
shear_modulus	Shear modulus
xx_array	x cooordinate for the domain within a 2D array
yy_array	y cooordinate for the domain within a 2D array

Returns

The surface displacement field

5.25.1.2 compute_open_pipe_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.pipe.compute_open_pipe_displacement (
              open_pipe_x,
              open_pipe_y,
              open_pipe_depth_0,
              open_pipe_depth_1,
              open_pipe_depth_2,
              open_pipe_radius,
              poisson_ratio,
              pressurization,
              shear_modulus,
              xx_array,
              yy_array )
```

Compute the surface displacements for an open pipe.

Parameters

open_pipe_x	x cooordinate for the pipe's center
open_pipe_y	y cooordinate for the pipe's center
open_pipe_depth←↩_0	Pipe's top depth with minimal pressurization
open_pipe_depth←↩_1	Pipe's top depth with maximal pressurization
open_pipe_depth←↩_2	Pipe's bottom depth
open_pipe_radius	Pipe's radius
poisson_ratio	Poisson's ratio
pressurization	Change of pressure applied to the pipe
shear_modulus	Shear modulus
xx_array	x cooordinate for the domain within a 2D array
yy_array	y cooordinate for the domain within a 2D array

Returns

The surface displacement field

5.26 pyinsar.processing.deformation.elastic_halfspace.surface_load Namespace Reference

Functions

- def compute_uniform_disk_load_displacement (disk_x, disk_y, disk_radius, poisson_ratio, pressure, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for a uniform disk load.

5.26.1 Function Documentation

5.26.1.1 compute_uniform_disk_load_displacement()

```
def pyinsar.processing.deformation.elastic_halfspace.surface_load.compute_uniform_disk_load_↩
displacement (
              disk_x,
              disk_y,
              disk_radius,
              poisson_ratio,
              pressure,
              shear_modulus,
              xx_array,
              yy_array )
```

Compute the surface displacements for a uniform disk load.

Parameters

disk_x	x cooordinate for the disk's center
disk_y	y cooordinate for the disk's center
disk_radius	Disk's radius
poisson_ratio	Poisson's ratio
pressure	Pressure applied by the disk
shear_modulus	Shear modulus
xx_array	x cooordinate for the domain within a 2D array
yy_array	y cooordinate for the domain within a 2D array

Returns

The surface displacement field

5.27 pyinsar.processing.discovery Namespace Reference

Namespaces

- classify_cnn

- coherence
- coregister
- deburst
- deformation_to_phase
- flat_earth
- interferogram
- los_deformation
- rotate_squares
- shown_cnn_classes
- temporal_decorrelation
- train_cnn
- wrap_phase

5.28 pyinsar.processing.discovery.classify_cnn Namespace Reference

Classes

- class ClassifyCNN

  Train a CNN.

5.29 pyinsar.processing.discovery.coherence Namespace Reference

Classes

- class Coherence

  Calculate coherence between single-look complex SAR images.

5.30 pyinsar.processing.discovery.coregister Namespace Reference

Classes

- class Coregister

  ∗∗∗ In Devolopment ∗∗∗ Pipeline item to coregister images

5.31 pyinsar.processing.discovery.deburst Namespace Reference

Classes

- class Deburst

  Debursts Sentinel-1 TOPSAR data.

5.32 pyinsar.processing.discovery.deformation_to_phase Namespace Reference

Classes

- class DeformationToPhase

  Convert deformation to phas.

5.33 pyinsar.processing.discovery.flat_earth Namespace Reference

Classes

- class FlatEarth

  ∗∗∗ *In Development* ∗∗∗ *Remove flat Earth contribution from interferogram*

5.34 pyinsar.processing.discovery.interferogram Namespace Reference

Classes

- class Interferogram

  Create Inteferogram from SLC data.

5.35 pyinsar.processing.discovery.los_deformation Namespace Reference

Classes

- class LOS_Deformation

  ∗∗∗ *In Development* ∗∗∗

5.36 pyinsar.processing.discovery.rotate_squares Namespace Reference

Classes

- class RotateSquares

  Generate new images by rotating subsections of data defined by Shapely squares.

Functions

- def rotateSquare (image, square, angle, order)

  Rotate a subsection of an image defined by a shapely square.

5.36.1 Function Documentation

5.36.1.1 rotateSquare()

```
def pyinsar.processing.discovery.rotateSquare (
              image,
              square,
              angle,
              order )
```

Rotate a subsection of an image defined by a shapely square.

Parameters

image	Full image containing subsection to be rotated
square	Shapely square
angle	Angle of rotation
order	Order of spline interpolation

5.37 pyinsar.processing.discovery.shown_cnn_classes Namespace Reference

Classes

- class ShowCNNClasses

  Dispay CNN Classifications on segments of an image.

5.38 pyinsar.processing.discovery.temporal_decorrelation Namespace Reference

Classes

- class TemporalDecorrelation

  Pipeline item to add temporal decorrelation to some phase.

5.39 pyinsar.processing.discovery.train_cnn Namespace Reference

Classes

- class TrainCNN

  Train a CNN.

5.40 pyinsar.processing.discovery.wrap_phase Namespace Reference

Classes

- class WrapPhase

  Pipeline Item that wraps phase.

5.41 pyinsar.processing.geography Namespace Reference

Namespaces

- coordinates
- geodesy
- geomorphometry

5.42 pyinsar.processing.geography.coordinates Namespace Reference

Functions

- def transform_to_pixel_coordinates (x, y, x_min, x_max, y_min, y_max, array_width, array_height)

  Array coordinates.
- def transform_to_geographic_coordinates (u, v, x_min, x_max, y_min, y_max, array_width, array_height)

  Transform some pixel coordinates in an array to geographic coordinates.
- def compute_x_and_y_coordinates_maps (x_min, x_max, y_min, y_max, array_width, array_height)

  Compute an array of x and y coordinates based on an extent and array shape.
- def extract_subgeoarray (georaster_array, georaster_extent, x_min, x_max, y_min, y_max, center_extent=False)
- def sample_nd_array (array, subarray_shape, steps=(1, 1))
- def sample_2d_array (array, subarray_shape, steps=(1, 1), is_shape_centered=False)
- def sample_2d_multiarray (array, subarray_shape, steps=(1, 1))
- def reproject_point (lon, lat, old_projection_EPSG=None, old_projection_wkt=None, old_projection_utm=None, new_projection_EPSG=None, new_projection_wkt=None, new_projection_utm=None)

  Projection.
- def find_utm_area (longitude, latitude)

  Find the UTM code and hemisphere from the longitude and latitude of a point.
- def reproject_georaster (georaster, new_cell_sizes, new_projection_EPSG=None, new_projection_wkt=None, new_projection_utm=None, new_extent=None, interpolation_method=gdal.GRA_Cubic, file_type='MEM', file_path='', data_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0., options=[ ])

  Change the projection of a GDAL georaster.
- def georaster_vertical_datum_shift (georaster, old_datum_proj4='+proj=longlat+datum=WGS84+no_defs+geoidgrids=egm96_15.gtx', new_datum_proj4='+proj=longlat+datum=WGS84+no_defs', file_type='MEM', file_path='', data_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0.)

Variables

- nopython

    Extract all the possible sub-arrays that do not contain any NaN.
- True
- nogil
- parallel

5.42.1 Function Documentation

5.42.1.1 compute_x_and_y_coordinates_maps()

```
def pyinsar.processing.geography.coordinates.compute_x_and_y_coordinates_maps (
              x_min,
              x_max,
              y_min,
              y_max,
              array_width,
              array_height )
```

Compute an array of x and y coordinates based on an extent and array shape.

Parameters

x_min	Minimal coordinate along the x axis (along the cell border)
x_max	Maximal coordinate along the x axis (along the cell border)
y_min	Minimal coordinate along the y axis (along the cell border)
y_max	Maximal coordinate along the y axis (along the cell border)
array_width	Width of the array (i.e., along the x axis)
array_height	Height of the array (i.e., along the y axis)

Returns

The coordinates' arrays

5.42.1.2 extract_subgeoarray()

```
def pyinsar.processing.geography.coordinates.extract_subgeoarray (
              georaster_array,
              georaster_extent,
              x_min,
              x_max,
              y_min,
              y_max,
              center_extent = False )
```

5.42.1.3 find_utm_area()

```
def pyinsar.processing.geography.coordinates.find_utm_area (
              longitude,
              latitude )
```

Find the UTM code and hemisphere from the longitude and latitude of a point.

Parameters

longitude	A float for the longitude
latitude	A float for the latitude

Returns

A tuple with the code of the UTM zone and the hemisphere (1: northern hemisphere; 0: southern hemisphere)

5.42.1.4 georaster_vertical_datum_shift()

```
def pyinsar.processing.geography.coordinates.georaster_vertical_datum_shift (
              georaster,
              old_datum_proj4 = '+proj=longlat +datum=WGS84 +no_defs +geoidgrids=egm96_15.gtx',
              new_datum_proj4 = '+proj=longlat +datum=WGS84 +no_defs',
              file_type = 'MEM',
              file_path = '',
              data_type = gdal.GDT_Float64,
              no_data_value = -99999.,
              scale = 1.,
              offset = 0.  )
```

5.42.1.5 reproject_georaster()

```
def pyinsar.processing.geography.coordinates.reproject_georaster (
              georaster,
              new_cell_sizes,
              new_projection_EPSG = None,
              new_projection_wkt = None,
              new_projection_utm = None,
              new_extent = None,
              interpolation_method = gdal.GRA_Cubic,
              file_type = 'MEM',
              file_path = '',
              data_type = gdal.GDT_Float64,
              no_data_value = -99999.,
              scale = 1.,
              offset = 0.,
              options = [] )
```

Change the projection of a GDAL georaster.

Parameters

georaster	The GDAL georaster
new_cell_sizes	Sizes (x, y) for cells of the georaster in the new projection
new_projection_EPSG	EPSG code of the new projection
new_projection_wkt	WKT code of the new projection (can be used instead of the new_projection_EPSG)
new_projection_utm	Tuple with the UTM zone code and if it's northern or not
new_extent	Tuple with the minimal x, maximal x, minimal y, maximal y for the new georaster
interpolation_method	Interpolation method used during the projection
file_type	Type to save the file (default is memory)
file_path	Where to store the new georasterEPSG_code (default is memory)
data_type	Data type of the georaster
no_data_value	No data value for the georaster
scale	Scaling factor for the georaster
offset	Offset factor for the georaster
options	List of options for compression

Returns

The GDAL georaster

5.42.1.6 reproject_point()

```
def pyinsar.processing.geography.coordinates.reproject_point (
              lon,
              lat,
              old_projection_EPSG = None,
              old_projection_wkt = None,
              old_projection_utm = None,
              new_projection_EPSG = None,
              new_projection_wkt = None,
              new_projection_utm = None )
```

Projection.

Reproject a single point

```
@param lon: Longitude of the point
@param lat: Latitude of the point
@param old_projection_EPSG: EPSG code of the old projection
@param old_projection_wkt: WKT code of the old projection (can be used instead
                           of the old_projection_EPSG)
@param old_projection_utm: Tuple with the UTM zone code and if it’s northern or not
@param new_projection_EPSG: EPSG code of the new projection
@param new_projection_wkt: WKT code of the new projection (can be used instead
                           of the new_projection_EPSG)
@param new_projection_utm: Tuple with the UTM zone code and if it’s northern or not

@return The coordinates’ arrays
```

5.42.1.7 sample_2d_array()

```
def pyinsar.processing.geography.coordinates.sample_2d_array (
              array,
              subarray_shape,
              steps = (1, 1),
              is_shape_centered = False )
```

5.42.1.8 sample_2d_multiarray()

```
def pyinsar.processing.geography.coordinates.sample_2d_multiarray (
              array,
              subarray_shape,
              steps = (1, 1) )
```

5.42.1.9 sample_nd_array()

```
def pyinsar.processing.geography.coordinates.sample_nd_array (
              array,
              subarray_shape,
              steps = (1, 1) )
```

5.42.1.10 transform_to_geographic_coordinates()

```
def pyinsar.processing.geography.coordinates.transform_to_geographic_coordinates (
              u,
              v,
              x_min,
              x_max,
              y_min,
              y_max,
              array_width,
              array_height )
```

Transform some pixel coordinates in an array to geographic coordinates.

Parameters

u	Pixel coordinate along the x axis to transform
v	Pixel coordinate along the y axis to transform
x_min	Minimal coordinate of the array along the x axis (along the cell border)
x_max	Maximal coordinate of the array along the x axis (along the cell border)
y_min	Minimal coordinate of the array along the y axis (along the cell border)
y_max	Maximal coordinate of the array along the y axis (along the cell border)
array_width	Width of the array (i.e., along the x axis)
array_height	Height of the array (i.e., along the y axis)

Returns

The geographic coordinates at the center of the pixel

5.42.1.11 transform_to_pixel_coordinates()

```
def pyinsar.processing.geography.coordinates.transform_to_pixel_coordinates (
              x,
              y,
              x_min,
              x_max,
              y_min,
              y_max,
              array_width,
              array_height )
```

Array coordinates.

Transform some geographic coordinates to pixel coordinates in an array

```
@param x: Coordinate along the x axis to transform
@param y: Coordinate along the y axis to transform
@param x_min: Minimal coordinate of the array along the x axis (along the cell border)
@param x_max: Maximal coordinate of the array along the x axis (along the cell border)
@param y_min: Minimal coordinate of the array along the y axis (along the cell border)
@param y_max: Maximal coordinate of the array along the y axis (along the cell border)
@param array_width: Width of the array (i.e., along the x axis)
@param array_height: Height of the array (i.e., along the y axis)

@return The pixel coordinates
```

5.42.2 Variable Documentation

5.42.2.1 nogil

```
pyinsar.processing.geography.coordinates.nogil
```

5.42.2.2 nopython

```
pyinsar.processing.geography.coordinates.nopython
```

Extract all the possible sub-arrays that do not contain any NaN.

Parameters

array	A 2D NumPy array
subarray_shape	The shape of the sub-arrays
steps	The step between each sub-array for each axis, to avoid sampling all the possible sub-arrays
is_shape_centered	True if the sub-arrays should be defined from their central cell, false if they should be defined from their top-left corner

Returns

The sub-arrays as a 3D NumPy array

Parameters

array	A 3D NumPy array. The first dimension represents the variables, the other two the x and y axis.
subarray_shape	The 2D shape of the sub-arrays
steps	The step between each sub-array for each axis, to avoid sampling all the possible sub-arrays

Returns

The sub-arrays as a 4D NumPy array

5.42.2.3 parallel

```
pyinsar.processing.geography.coordinates.parallel
```

5.42.2.4 True

```
pyinsar.processing.geography.coordinates.True
```

5.43 pyinsar.processing.geography.geodesy Namespace Reference

Functions

- def compute_great_circle_distance_and_bearing (rad_longitude_1, rad_latitude_1, rad_longitude_2, rad_latitude_2, planet_radius)

  Geodesy on a sphere.
- def compute_lonlat_from_distance_bearing (rad_longitude_1, rad_latitude_1, distance, rad_bearing, planet_radius)
- def direct_vincenty_formula (rad_lon_1, rad_lat_1, distance, rad_bearing_1, a, f, eps=1e-12)
- def direct_vincenty_formula_for_array (rad_longitude_1_array, rad_latitude_1_array, distance_array, rad_bearing_1, a, f, eps=1e-12)
- def update_lambda (Lambda, reduced_rad_lat_1, reduced_rad_lat_2, diff_lon, f)
- def inverse_vincenty_formula (rad_lon_1, rad_lat_1, rad_lon_2, rad_lat_2, a, f, eps=1e-12, max_iter=200)
- def inverse_vincenty_formula_for_array (rad_longitude_1, rad_latitude_1, rad_longitude_2_array, rad_latitude_2_array, a, f, eps=1e-12, max_iter=200)
- def compute_point_to_line_distance_on_ellipsoid (rad_point_lon, rad_point_lat, rad_geodesic_origin_lon, rad_geodesic_origin_lat, rad_geodesic_bearing, a, f, eps=1e-12, max_iter=200)
- def compute_point_to_line_distance_for_array (rad_longitude_1, rad_latitude_1, rad_longitude_2_array, rad_latitude_2_array, rad_bearing, a, f, eps=1e-12, max_iter=200)

Variables

- nopython

  Geodesy on an oblate spheroid.

5.43.1 Function Documentation

5.43.1.1 compute_great_circle_distance_and_bearing()

```
def pyinsar.processing.geography.geodesy.compute_great_circle_distance_and_bearing (
              rad_longitude_1,
              rad_latitude_1,
              rad_longitude_2,
              rad_latitude_2,
              planet_radius )
```

Geodesy on a sphere.

Compute the distance and initial bearing between two points on a sphere

```
@param rad_longitude_1: Longitude of the first point (in radian)
@param rad_latitude_1: Latitude of the first point (in radian)
@param rad_longitude_2: Longitude of the second point (in radian)
@param rad_latitude_2: Latitude of the second point (in radian)
@param planet_radius: Radius of the considered planet (same unit as the distance)

@return The Haversine distance and the initial bearing (in radian)
```

5.43.1.2 compute_lonlat_from_distance_bearing()

```
def pyinsar.processing.geography.geodesy.compute_lonlat_from_distance_bearing (
              rad_longitude_1,
              rad_latitude_1,
              distance,
              rad_bearing,
              planet_radius )
```

5.43.1.3 compute_point_to_line_distance_for_array()

```
def pyinsar.processing.geography.geodesy.compute_point_to_line_distance_for_array (
              rad_longitude_1,
              rad_latitude_1,
              rad_longitude_2_array,
              rad_latitude_2_array,
              rad_bearing,
              a,
              f,
              eps = 1e-12,
              max_iter = 200 )
```

5.43.1.4 compute_point_to_line_distance_on_ellipsoid()

```
def pyinsar.processing.geography.geodesy.compute_point_to_line_distance_on_ellipsoid (
              rad_point_lon,
              rad_point_lat,
              rad_geodesic_origin_lon,
              rad_geodesic_origin_lat,
              rad_geodesic_bearing,
              a,
              f,
              eps = 1e-12,
              max_iter = 200 )
```

5.43.1.5 direct_vincenty_formula()

```
def pyinsar.processing.geography.geodesy.direct_vincenty_formula (
              rad_lon_1,
              rad_lat_1,
              distance,
              rad_bearing_1,
              a,
              f,
              eps = 1e-12 )
```

5.43.1.6 direct_vincenty_formula_for_array()

```
def pyinsar.processing.geography.geodesy.direct_vincenty_formula_for_array (
              rad_longitude_1_array,
              rad_latitude_1_array,
              distance_array,
              rad_bearing_1,
              a,
              f,
              eps = 1e-12 )
```

5.43.1.7 inverse_vincenty_formula()

```
def pyinsar.processing.geography.geodesy.inverse_vincenty_formula (
              rad_lon_1,
              rad_lat_1,
              rad_lon_2,
              rad_lat_2,
              a,
              f,
              eps = 1e-12,
              max_iter = 200 )
```

5.43.1.8 inverse_vincenty_formula_for_array()

```
def pyinsar.processing.geography.geodesy.inverse_vincenty_formula_for_array (
              rad_longitude_1,
              rad_latitude_1,
              rad_longitude_2_array,
              rad_latitude_2_array,
              a,
              f,
              eps = 1e-12,
              max_iter = 200 )
```

5.43.1.9 update_lambda()

```
def pyinsar.processing.geography.geodesy.update_lambda (
              Lambda,
              reduced_rad_lat_1,
              reduced_rad_lat_2,
              diff_lon,
              f )
```

5.43.2 Variable Documentation

5.43.2.1 nopython

```
pyinsar.processing.geography.geodesy.nopython
```

Geodesy on an oblate spheroid.

Update the parameter lambda of Vincenty's inverse formula.

5.44 pyinsar.processing.geography.geomorphometry Namespace Reference

Functions

- def add_symmetric_border (array, border_size=1)
- def compute_gradient_at_cell (array, j, i, grid_yx_spacing, axis=1)
- def compute_horne_slope (array, grid_yx_spacing)

Variables

- nopython

  Add a symmetric border to a 2D array.

5.44.1 Function Documentation

5.44.1.1 add_symmetric_border()

```
def pyinsar.processing.geography.geomorphometry.add_symmetric_border (
              array,
              border_size = 1 )
```

5.44.1.2 compute_gradient_at_cell()

```
def pyinsar.processing.geography.geomorphometry.compute_gradient_at_cell (
              array,
              j,
              i,
              grid_yx_spacing,
              axis = 1 )
```

5.44.1.3 compute_horne_slope()

```
def pyinsar.processing.geography.geomorphometry.compute_horne_slope (
              array,
              grid_yx_spacing )
```

5.44.2 Variable Documentation

5.44.2.1 nopython

```
pyinsar.processing.geography.geomorphometry.nopython
```

Add a symmetric border to a 2D array.

Compute Horn's slope of a 2D array with a fixed cell size.

Compute Horn's gradient for a given cell of an array.

Parameters

array	The array
border_size	The size of the border

Returns

The expended array

Parameters

array	The array
j	The index of the cell along the y axis
i	The index of the cell along the x axis
grid_yx_spacing	The cell size, which is considered fixed for the entire array
axis	the axis along which the gradient is computed (0: y; 1: x)

Returns

The gradient value for the cell

Parameters

array	The array
grid_yx_spacing	The cell size, which is considered fixed for the entire array

Returns

The slope (in degree)

5.45 pyinsar.processing.instruments Namespace Reference

Namespaces

- sentinel

5.46 pyinsar.processing.instruments.sentinel Namespace Reference

Classes

- class RampPolynomial

  Polynomial used for quantities relating to deramping sentinel.
- class SentinelRamp

  Calcuate the combined ramp and modulated phase in Sentinel.

Functions

- def transform_slc (slc, deramped_phase, transformation_matrix)
- def find_overlapping_valid_lines (metadata_tree)

    Determine which lines between bursts overlap.
- def get_valid_lines (metadata_tree, per_burst=False)

    Retrieve all lines that contain some valid data.
- def select_valid_lines (data, tree, cut=True)

    Extract burst information from SLC.
- def retrieve_azimuth_time (in_tree)

    Retrieves the zero azimuth time for all the lines in the data.
- def read_geolocation (tree)

    Read in geolocation data.
- def update_geolocation_lines (tree, azimuth_times, geolocation_data)

    Update which line is associated with geolocation data using azimuth times.
- def get_sentinel_extents (geolocation, offset=0.0)

    Get the extents (latitude and longitude) of a sentinel-1 image given its geolocation information.

5.46.1 Function Documentation

5.46.1.1 find_overlapping_valid_lines()

```
def pyinsar.processing.instruments.sentinel.find_overlapping_valid_lines (
              metadata_tree )
```

Determine which lines between bursts overlap.

Parameters

metadata_tree	Sentinel metadata XML tree

Returns

List of overlapping index ranges

5.46.1.2 get_sentinel_extents()

```
def pyinsar.processing.instruments.sentinel.get_sentinel_extents (
              geolocation,
              offset = 0.0 )
```

Get the extents (latitude and longitude) of a sentinel-1 image given its geolocation information.

Parameters

geolocation	Geolocation data read in by read_geolocation
offset	Extra offset to add to the extent

Returns

Latitude and longitude extents of a sentinel-1

5.46.1.3 get_valid_lines()

```
def pyinsar.processing.instruments.sentinel.get_valid_lines (
              metadata_tree,
              per_burst = False )
```

Retrieve all lines that contain some valid data.

Parameters

metadata_tree	Sentinel XML metadata tree
per_burst	Retrieve the burst data as seperate arrays

Returns

Sentinel data for all lines that are valid

5.46.1.4 read_geolocation()

```
def pyinsar.processing.instruments.sentinel.read_geolocation (
              tree )
```

Read in geolocation data.

Parameters

tree	Sentinel metadata as an ElementTree

Returns

Geolocation metadata

5.46.1.5 retrieve_azimuth_time()

```
def pyinsar.processing.instruments.sentinel.retrieve_azimuth_time (
              in_tree )
```

Retrieves the zero azimuth time for all the lines in the data.

Parameters

in_tree	SLC Metadata as an ElementTree

Returns

Pandas series of azimuth times for each line

5.46.1.6 select_valid_lines()

```
def pyinsar.processing.instruments.sentinel.select_valid_lines (
              data,
              tree,
              cut = True )
```

Extract burst information from SLC.

Parameters

data	Input SLC data
tree	Metadata as an ElementTree
cut	Remove invalid lines

Returns

A list containing individual images of each burst

5.46.1.7 transform_slc()

```
def pyinsar.processing.instruments.sentinel.transform_slc (
              slc,
              deramped_phase,
              transformation_matrix )
```

Parameters

slc	Input slc
deramped_phase	Phase to be removed before the transformation and to be readded afterwards
transformation_matrix	A 2x3 transformation matrix to be used by warpAffine by opencv

Returns

transformed slc

5.46.1.8 update_geolocation_lines()

```
def pyinsar.processing.instruments.sentinel.update_geolocation_lines (
              tree,
              azimuth_times,
              geolocation_data )
```

Update which line is associated with geolocation data using azimuth times.

Parameters

tree	Sentinel XML metadata
azimuth_times	Azimuth times
geolocation_data	Geolocation data read in by read_geolocation

Returns

New lines for the geolocation data

5.47 pyinsar.processing.isce Namespace Reference

Namespaces

- input_file

5.48 pyinsar.processing.isce.input_file Namespace Reference

Functions

- def create_product_xml (xml_path, product_path, product_type='master', product_output_path=None, product↩_orbit_path=None, product_auxiliary_data_path=None, do_add=True)

Create the xml file defining a Sentinel-1 product for processing with ISCE.

- def create_topsApp_xml (xml_folder_path, master_path, slave_path, master_output_path=None, slave_output_path=None, master_orbit_path=None, slave_orbit_path=None, master_auxiliary_data_path=None, slave_auxiliary_data_path=None, do_unwrap=True, unwrapper_name='snaphu_mcf', xml_filename='topsApp.xml')

  Create the topsApp.xml file for processing Sentinel-1 data with ISCE.
- def prepare_topsApps (product_paths, result_folder_path, orbit_path=None, auxiliary_data_path=None, do_unwrap=True, unwrapper_name='snaphu_mcf')

5.48.1 Function Documentation

5.48.1.1 create_product_xml()

```
def pyinsar.processing.isce.input_file.create_product_xml (
              xml_path,
              product_path,
              product_type = 'master',
              product_output_path = None,
              product_orbit_path = None,
              product_auxiliary_data_path = None,
              do_add = True )
```

Create the xml file defining a Sentinel-1 product for processing with ISCE.

Parameters

xml_path	Path to the xml file
product_path	Path to the Sentinel-1 product
product_type	Master or slave product
product_output_path	Path for the processing outputs of this product
product_orbit_path	Path to the folder containing orbit files
product_auxiliary_data_path	Path to the folder containing auxiliary data
do_add	True if the component is added to an already existing xml file, false otherwise

5.48.1.2 create_topsApp_xml()

```
def pyinsar.processing.isce.input_file.create_topsApp_xml (
              xml_folder_path,
              master_path,
              slave_path,
              master_output_path = None,
              slave_output_path = None,
              master_orbit_path = None,
              slave_orbit_path = None,
              master_auxiliary_data_path = None,
              slave_auxiliary_data_path = None,
              do_unwrap = True,
              unwrapper_name = 'snaphu_mcf',
              xml_filename = 'topsApp.xml' )
```

Create the topsApp.xml file for processing Sentinel-1 data with ISCE.

Parameters

xml_folder_path	Path to the folder that will contain the xml file
master_path	Path to the master Sentinel-1 product
slave_path	Path to the slave Sentinel-1 product
master_output_path	Path for the processing outputs of the master product
slave_output_path	Path for the processing outputs of the slave product
master_orbit_path	Path to the folder containing orbit files for the master product
slave_orbit_path	Path to the folder containing orbit files for the slave product
master_auxiliary_data_path	Path to the folder containing auxiliary data for the master product
slave_auxiliary_data_path	Path to the folder containing auxiliary data for the slave product
do_unwrap	True to unwrap the created interferogram, false otherwise
unwrapper_name	Name of the unwrapper when do_unwrap is true
xml_filename	Name of the topsApp.xml file to create

Returns

The path to the created topsApp.xml file

5.48.1.3 prepare_topsApps()

```
def pyinsar.processing.isce.input_file.prepare_topsApps (
              product_paths,
              result_folder_path,
              orbit_path = None,
              auxiliary_data_path = None,
              do_unwrap = True,
              unwrapper_name = 'snaphu_mcf' )
```

5.49 pyinsar.processing.machine_learning Namespace Reference

Namespaces

- geostatistics

5.50 pyinsar.processing.machine_learning.geostatistics Namespace Reference

Namespaces

- direct_sampling
- geostatistics_utils
- sequential_gaussian_simulation
- variogram

5.51 pyinsar.processing.machine_learning.geostatistics.direct_sampling Namespace Reference

Functions

- def compute_neighborhood_lag_vectors (neighborhood_shape, grid_yx_spacing, delta)
- def compute_neighborhoods (simulation_array, data_weight_array, cell_j, cell_i, lag_vectors, lag_distances, max_number_data, max_density_data, neighborhood_shape, rotation_angle_rad, scaling_factor, no_data_value)
- def compute_continuous_distance (training_image_array, ti_j, ti_i, ti_ranges_max, neighbor_indexes, neighbor_values, neighbor_numbers, min_distances, var_k, max_non_matching_proportion, no_data_value)
- def compute_discrete_distance (training_image_array, ti_j, ti_i, neighbor_indexes, neighbor_values, neighbor_numbers, min_distances, var_k, max_non_matching_proportion, no_data_value)
- def find_closest_cell_in_training_image (training_image_array, ti_ranges_max, ti_indices, ti_index, neighbor_indexes, neighbor_values, neighbor_numbers, distance_thresholds, max_non_matching_proportion, ti_fraction, no_data_value)
- def prepare_training_image (array, variable_types)
- def is_any_equal (list_1, value)
- def is_any_nan (list_1)
- def run_ds (data_array, training_image_array, variable_types, distance_thresholds, ti_fraction, max_number_data, max_density_data, neighborhood_shape=(math.inf, math.inf), grid_yx_spacing=(1., 1.), delta=0., conditioning_data_weight=1., max_non_matching_proportion=1., start_parameter_reduction=1, reduction_factor=1, rotation_angle_array=np.empty((1, 1)), scaling_factor_array=np.empty((1, 1, 1)), number_postproc=0, postproc_factor=1, number_realizations=1, path_type=PathType.RANDOM, seed=100, no_data_value=-99999)
- def simulate_ds_realization (data_array, data_weight_array, training_image_array, ti_ranges_max, ti_indices, distance_thresholds, ti_fraction, max_number_data, max_density_data, lag_vectors, lag_distances, neighborhood_shape, max_non_matching_proportion, start_parameter_reduction, reduction_factor, rotation_angle_array, scaling_factor_array, number_postproc, postproc_factor, path_type, seed, no_data_value)
- def run_parallel_ds (data_array, training_image_array, variable_types, distance_thresholds, ti_fraction, max_number_data, max_density_data, neighborhood_shape=(math.inf, math.inf), grid_yx_spacing=(1., 1.), delta=0., conditioning_data_weight=1., max_non_matching_proportion=1., start_parameter_reduction=1, reduction_factor=1, rotation_angle_array=np.empty((1, 1)), scaling_factor_array=np.empty((1, 1, 1)), number_postproc=0, postproc_factor=1, number_realizations=1, path_type=PathType.RANDOM, seed=100, no_data_value=-99999)

Variables

- nopython

  Compute the lag vectors for the neighborhood, assuming a regular grid.

5.51.1 Function Documentation

5.51.1.1 compute_continuous_distance()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_continuous_distance
(
              training_image_array,
              ti_j,
              ti_i,
              ti_ranges_max,
              neighbor_indexes,
              neighbor_values,
              neighbor_numbers,
              min_distances,
              var_k,
              max_non_matching_proportion,
              no_data_value )
```

5.51.1.2 compute_discrete_distance()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_discrete_distance (
              training_image_array,
              ti_j,
              ti_i,
              neighbor_indexes,
              neighbor_values,
              neighbor_numbers,
              min_distances,
              var_k,
              max_non_matching_proportion,
              no_data_value )
```

5.51.1.3 compute_neighborhood_lag_vectors()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_neighborhood_lag_↩
vectors (
              neighborhood_shape,
              grid_yx_spacing,
              delta )
```

5.51.1.4 compute_neighborhoods()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_neighborhoods (
              simulation_array,
              data_weight_array,
              cell_j,
              cell_i,
              lag_vectors,
              lag_distances,
              max_number_data,
              max_density_data,
              neighborhood_shape,
              rotation_angle_rad,
              scaling_factor,
              no_data_value )
```

5.51.1.5 find_closest_cell_in_training_image()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.find_closest_cell_in_↩
training_image (
              training_image_array,
              ti_ranges_max,
              ti_indices,
              ti_index,
              neighbor_indexes,
              neighbor_values,
              neighbor_numbers,
              distance_thresholds,
              max_non_matching_proportion,
              ti_fraction,
              no_data_value )
```

5.51.1.6 is_any_equal()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.is_any_equal (
              list_1,
              value )
```

5.51.1.7 is_any_nan()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.is_any_nan (
              list_1 )
```

5.51.1.8 prepare_training_image()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.prepare_training_image (
              array,
              variable_types )
```

5.51.1.9 run_ds()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.run_ds (
              data_array,
              training_image_array,
              variable_types,
              distance_thresholds,
              ti_fraction,
              max_number_data,
              max_density_data,
              neighborhood_shape = (math.inf, math.inf),
              grid_yx_spacing = (1., 1.),
              delta = 0.,
              conditioning_data_weight = 1.,
              max_non_matching_proportion = 1.,
              start_parameter_reduction = 1,
              reduction_factor = 1,
              rotation_angle_array = np.empty((1, 1)),
              scaling_factor_array = np.empty((1, 1, 1)),
              number_postproc = 0,
              postproc_factor = 1,
              number_realizations = 1,
              path_type = PathType.RANDOM,
              seed = 100,
              no_data_value = -99999 )
```

5.51.1.10 run_parallel_ds()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.run_parallel_ds (
              data_array,
              training_image_array,
              variable_types,
              distance_thresholds,
              ti_fraction,
              max_number_data,
              max_density_data,
              neighborhood_shape = (math.inf, math.inf),
              grid_yx_spacing = (1., 1.),
              delta = 0.,
              conditioning_data_weight = 1.,
              max_non_matching_proportion = 1.,
              start_parameter_reduction = 1,
              reduction_factor = 1,
              rotation_angle_array = np.empty((1, 1)),
              scaling_factor_array = np.empty((1, 1, 1)),
              number_postproc = 0,
              postproc_factor = 1,
              number_realizations = 1,
              path_type = PathType.RANDOM,
              seed = 100,
              no_data_value = -99999 )
```

5.51.1.11 simulate_ds_realization()

```
def pyinsar.processing.machine_learning.geostatistics.direct_sampling.simulate_ds_realization (
              data_array,
              data_weight_array,
              training_image_array,
              ti_ranges_max,
              ti_indices,
              distance_thresholds,
              ti_fraction,
              max_number_data,
              max_density_data,
              lag_vectors,
              lag_distances,
              neighborhood_shape,
              max_non_matching_proportion,
              start_parameter_reduction,
              reduction_factor,
              rotation_angle_array,
              scaling_factor_array,
              number_postproc,
              postproc_factor,
              path_type,
              seed,
              no_data_value )
```

5.51.2 Variable Documentation

5.51.2.1 nopython

```
pyinsar.processing.machine_learning.geostatistics.direct_sampling.nopython
```

Compute the lag vectors for the neighborhood, assuming a regular grid.

Check if there is any NaN in a list (or tuple, or 1D NumPy array)

Check if there is a given value in a list (or tuple, or 1D NumPy array)

Compute the distance between two neighborhoods for a discrete variable.

Compute the distance between two neighborhoods for a continuous variable.

Parameters

neighborhood_shape	The maximal coverage of the neighborhood along each axis
grid_yx_spacing	The cell size along each axis (y, x)
delta	A weight for the neighboring cells during simulation, a high delta giving more influence to the cells closer to the cell to simulate

Returns

The closest cells to the center cell of the neighborhood and the corresponding weighted distance

Parameters

training_image_array	A NumPy array containing the training image, from which the simulated values are borrowed. It should be a 3D array, with one dimension for the variable(s), and two spatial dimensions
ti_j	Index along the y axis of the initial cell to visit in the training image
ti_i	Index along the x axis of the initial cell to visit in the training image
ti_ranges_max	Squared difference between the min and max value of each variable
neighbor_indexes	Indexes of the neighborhood from the cell to simulate
neighbor_values	Values of the neighborhood in the simulation grid
neighbor_numbers	Number of neighbors for each variable
min_distances	The minimal distance of each variable found so far
var_k	Index of the variable
max_non_matching_proportion	Authorized proportion of non-matching nodes, i.e., whose distance is below the threshold for the variable
no_data_value	The no-data value, which defines the cell to simulate

Returns

The distance

Parameters

training_image_array	A NumPy array containing the training image, from which the simulated values are borrowed. It should be a 3D array, with one dimension for the variable(s), and two spatial dimensions
ti_j	Index along the y axis of the initial cell to visit in the training image
ti_i	Index along the x axis of the initial cell to visit in the training image

Parameters

neighbor_indexes	Indexes of the neighborhood from the cell to simulate
neighbor_values	Values of the neighborhood in the simulation grid
neighbor_numbers	Number of neighbors for each variable
min_distances	The minimal distance of each variable found so far
var_k	Index of the variable
max_non_matching_proportion	Authorized proportion of non-matching nodes, i.e., whose distance is below the threshold for the variable
no_data_value	The no-data value, which defines the cell to simulate

Returns

The distance

Parameters

list←↩ _1	The list
value	The value

Returns

True if there is the value, false otherwise

Parameters

list←↩ _1	The list

Returns

True if there is a NaN, false otherwise

5.52 pyinsar.processing.machine_learning.geostatistics.geostatistics_utils Namespace Reference

Classes

- class PathType
- class VariableType

Functions

- def unflatten_index (flattened_index, array_shape)
- def standardize (x)

  Reduce and center a float or array.
- def normalize (x)

  Reduce and center a float or array.

Variables

- nopython

  Unflatten an index for a 2D array.
- True
- nogil

5.52.1 Function Documentation

5.52.1.1 normalize()

```
def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.normalize (
              x )
```

Reduce and center a float or array.

Parameters

x	The float or array

Returns

A float or array

5.52.1.2 standardize()

```
def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.standardize (
              x )
```

Reduce and center a float or array.

Parameters

x	The float or array

Returns

A float or array

5.52.1.3 unflatten_index()

```
def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.unflatten_index (
              flattened_index,
              array_shape )
```

5.52.2 Variable Documentation

5.52.2.1 nogil

```
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.nogil
```

5.52.2.2 nopython

```
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.nopython
```

Unflatten an index for a 2D array.

Parameters

flattened_index	The flattened index (i.e., a single integer)
array_shape	The shape of the array for the two dimensions (j, i)

Returns

The 2D index (j, i)

5.52.2.3 True

```
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.True
```

5.53 pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation Namespace Reference

Classes

- class KrigingMethod

Functions

- def merge_secondary_data (secondary_data_array, correlations_with_primary, correlations_between_↩ secondary)

  Merging secondary data.
- def compute_euclidean_distance (cell_1, cell_2)
- def compute_axis_aligned_ellipse_range (neighborhood_range, neighborhood_azimuth_rad)
- def compute_axis_aligned_neighborhood_shape (neighborhood_range, neighborhood_azimuth, grid_yx_↩ spacing)
- def compute_neighborhood_template (neighborhood_range, grid_yx_spacing, vario_models, vario_sills, vario↩ _ranges, vario_azimuth_rad, rotation_matrix, eps=0.0001)
- def get_neighborhood (cell_index, simulation_array, neighborhood_template, max_number_data, no_data_value)
- def get_values_matrix (neighborhood, simulation_array)
- def get_data_to_data_matrix (kriging_method, cell_index, neighborhood, correlation_template, secondary_↩ data_weight)
- def get_data_to_unknown_matrix (kriging_method, cell_index, neighborhood, correlation_template, secondary↩ _data_weight)
- def solve_kriging_system (cell_index, neighborhood, simulation_array, primary_mean, primary_variance, correlation_template, secondary_data_weight, secondary_data_mean, secondary_data_array)
- def run_sgs (data_array, grid_yx_spacing, vario_models, vario_sills, vario_azimuth, vario_ranges, number↩ _realizations=1, path_type=PathType.RANDOM, kriging_method=KrigingMethod.SIMPLE, neighborhood_↩ range=(math.nan, math.nan), max_number_data=12, secondary_data_weight=math.nan, secondary_data_↩ array=np.empty((1, 1)), seed=100, no_data_value=-99999.)
- def simulate_sgs_realization (data_array, path_type, primary_mean, primary_variance, neighborhood_template, correlation_template, max_number_data, secondary_data_weight, secondary_data_array, seed, no_data_value)
- def run_parallel_sgs (data_array, grid_yx_spacing, vario_models, vario_sills, vario_azimuth, vario_↩ ranges, number_realizations=1, path_type=PathType.RANDOM, kriging_method=KrigingMethod.SIMP↩ LE, neighborhood_range=(math.nan, math.nan), max_number_data=12, secondary_data_weight=math.nan, secondary_data_array=np.empty((1, 1)), seed=100, nb_threads=4, no_data_value=-99999.)
- def inverse_standard_normal_cdf (x)

  Data transform.
- def compute_averaged_cumulative_distribution_from_array (value_array)
- def normal_score_tranform (value_array)

  Transform the values of an array to a normal distribution.

Variables

- nopython

  Sequential Gaussian Simulation (SGS)
- True
- nogil

5.53.1 Function Documentation

5.53.1.1 compute_averaged_cumulative_distribution_from_array()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_↩
averaged_cumulative_distribution_from_array (
              value_array )
```

5.53.1.2 compute_axis_aligned_ellipse_range()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_↩
axis_aligned_ellipse_range (
              neighborhood_range,
              neighborhood_azimuth_rad )
```

5.53.1.3 compute_axis_aligned_neighborhood_shape()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_↩
axis_aligned_neighborhood_shape (
              neighborhood_range,
              neighborhood_azimuth,
              grid_yx_spacing )
```

5.53.1.4 compute_euclidean_distance()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_↩
euclidean_distance (
              cell_1,
              cell_2 )
```

5.53.1.5 compute_neighborhood_template()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_↩
neighborhood_template (
              neighborhood_range,
              grid_yx_spacing,
              vario_models,
              vario_sills,
              vario_ranges,
              vario_azimuth_rad,
              rotation_matrix,
              eps = 0.0001 )
```

5.53.1.6 get_data_to_data_matrix()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_data_to↩
_data_matrix (
              kriging_method,
              cell_index,
              neighborhood,
              correlation_template,
              secondary_data_weight )
```

5.53.1.7 get_data_to_unknown_matrix()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_data_to↩
_unknown_matrix (
              kriging_method,
              cell_index,
              neighborhood,
              correlation_template,
              secondary_data_weight )
```

5.53.1.8 get_neighborhood()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_neighborhood
(
              cell_index,
              simulation_array,
              neighborhood_template,
              max_number_data,
              no_data_value )
```

5.53.1.9 get_values_matrix()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_values_↩
matrix (
              neighborhood,
              simulation_array )
```

5.53.1.10 inverse_standard_normal_cdf()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.inverse_↩
standard_normal_cdf (
              x )
```

Data transform.

Compute the inverse of a normal cumulative distribution

```
@param x: A float or array

@return The values normally distributed
```

5.53.1.11 merge_secondary_data()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.merge_↩
secondary_data (
              secondary_data_array,
              correlations_with_primary,
              correlations_between_secondary )
```

Merging secondary data.

Merge several secondary data (see Babak and Deutsch, 2009, doi:10.1016/j.petrol.2009.08.001)

```
@param secondary_data_array: A 3D array gathering several 2D secondary data
@param correlations_with_primary: Correlation weights between the main data
                                  and the secondary data
@param correlations_between_secondary: Correlation weights between the secondary data

@return A 2D array with the merged secondary data and a weight for the merged data
```

5.53.1.12 normal_score_tranform()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.normal_↩
score_tranform (
              value_array )
```

Transform the values of an array to a normal distribution.

Parameters

value_array	An array

Returns

An array with its values normally distributed

5.53.1.13 run_parallel_sgs()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.run_parallel↩
_sgs (
              data_array,
              grid_yx_spacing,
              vario_models,
              vario_sills,
              vario_azimuth,
              vario_ranges,
              number_realizations = 1,
              path_type = PathType.RANDOM,
              kriging_method = KrigingMethod.SIMPLE,
              neighborhood_range = (math.nan, math.nan),
              max_number_data = 12,
              secondary_data_weight = math.nan,
              secondary_data_array = np.empty((1, 1)),
              seed = 100,
              nb_threads = 4,
              no_data_value = -99999.  )
```

5.53.1.14 run_sgs()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.run_sgs (
              data_array,
              grid_yx_spacing,
              vario_models,
              vario_sills,
              vario_azimuth,
              vario_ranges,
              number_realizations = 1,
              path_type = PathType.RANDOM,
              kriging_method = KrigingMethod.SIMPLE,
              neighborhood_range = (math.nan, math.nan),
              max_number_data = 12,
              secondary_data_weight = math.nan,
              secondary_data_array = np.empty((1, 1)),
              seed = 100,
              no_data_value = -99999.  )
```

5.53.1.15 simulate_sgs_realization()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.simulate_↩
sgs_realization (
              data_array,
              path_type,
              primary_mean,
              primary_variance,
              neighborhood_template,
              correlation_template,
              max_number_data,
              secondary_data_weight,
              secondary_data_array,
              seed,
              no_data_value )
```

5.53.1.16 solve_kriging_system()

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.solve_↩
kriging_system (
              cell_index,
              neighborhood,
              simulation_array,
              primary_mean,
              primary_variance,
              correlation_template,
              secondary_data_weight,
              secondary_data_mean,
              secondary_data_array )
```

5.53.2 Variable Documentation

5.53.2.1 nogil

```
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.nogil
```

5.53.2.2 nopython

```
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.nopython
```

Sequential Gaussian Simulation (SGS)

Get the matrices of the kriging system and solve it.

Get the matrix of variogram values between the cells with already a value.

Get the matrix of already simulated values around the cell to estimate.

Compute the shape (in cells) of an ellipse along the y and x axes.

Compute the extent of an ellipse along the y and x axes.

Compute the 2D Euclidean distance

```
@param cell_1: The first point
@param array_shape: The second point

@return The distance
```

Parameters

neighborhood_range	The major and minor axis length of the ellipse
neighborhood_azimuth_rad	The azimuth giving the orientation of the major axis

Returns

The extent

Parameters

neighborhood_range	The major and minor axis length of the ellipse
neighborhood_azimuth_rad	The azimuth giving the orientation of the major axis (in radian)
grid_yx_spacing	The cell size along each axis (y, x)

Returns

The shape

Parameters

neighborhood	The data around the cell to estimate
simulation_array	The simulation grid

Returns

The matrix

Parameters

kriging_method	The kriging method to use (see KrigingMethod)
cell_index	The cell to estimate
neighborhood	The data around the point to estimate
correlation_template	The variogram values in the neighborhood
secondary_data_weight	The weight for the secondary data (math.nan if no secondary data)

Returns

The matrix

Parameters

cell_index	The cell to estimate
neighborhood	The data around the point to estimate
simulation_array	The simulation grid
primary_mean	Mean for the variable to estimate (if using a simple kriging, math.nan if using an ordinary kriging)
primary_variance	Variance for the variable to estimate
correlation_template	The variogram values in the neighborhood
secondary_data_weight	The weight for the secondary data (math.nan if no secondary data)
secondary_data_mean	Mean for the secondary data
secondary_data_array	The secondary data

Returns

The estimation of the mean and variance for the cell

5.53.2.3 True

```
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.True
```

5.54 pyinsar.processing.machine_learning.geostatistics.variogram Namespace Reference

Classes

- class VariogramModel

  2D theoretical variogram

Functions

- def compute_experimental_variogram (value_array, grid_yx_spacing, number_of_lags, lag_unit_distance, tolerance=None, sampling=1., no_data_value=-99999)
- def nugget_variogram (reduced_distance, variance_contribution)
- def gaussian_variogram (reduced_distance, variance_contribution)
- def spherical_variogram (reduced_distance, variance_contribution)
- def exponential_variogram (reduced_distance, variance_contribution)
- def compute_variogram (delta_y, delta_x, vario_models, vario_sills, vario_ranges, rotation_matrix)
- def vectorized_gaussian_variogram (distance, vario_range, variance_contribution)

  Vectorized theoretical variogram.
- def vectorized_spherical_variogram (distance, vario_range, variance_contribution)

  Compute the value of a variogram with a spherical model.
- def vectorized_exponential_variogram (distance, vario_range, variance_contribution)

  Compute the value of a variogram with an exponential model.
- def map_2D_variogram (vario_models, vario_sills, vario_azimuth, vario_ranges, neighborhood_range, map_←↩ shape, grid_spacing)
- def compute_range_variogram (deltas_y, deltas_x, vario_models, vario_sills, vario_ranges, vario_azimuth=0.)

Variables

- nopython

  2D experimental variogram
- True
- nogil

5.54.1 Function Documentation

5.54.1.1 compute_experimental_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.compute_experimental_variogram (
              value_array,
              grid_yx_spacing,
              number_of_lags,
              lag_unit_distance,
              tolerance = None,
              sampling = 1.,
              no_data_value = -99999 )
```

5.54.1.2 compute_range_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.compute_range_variogram (
              deltas_y,
              deltas_x,
              vario_models,
              vario_sills,
              vario_ranges,
              vario_azimuth = 0.  )
```

5.54.1.3 compute_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.compute_variogram (
              delta_y,
              delta_x,
              vario_models,
              vario_sills,
              vario_ranges,
              rotation_matrix )
```

5.54.1.4 exponential_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.exponential_variogram (
              reduced_distance,
              variance_contribution )
```

5.54.1.5 gaussian_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.gaussian_variogram (
              reduced_distance,
              variance_contribution )
```

5.54.1.6 map_2D_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.map_2D_variogram (
              vario_models,
              vario_sills,
              vario_azimuth,
              vario_ranges,
              neighborhood_range,
              map_shape,
              grid_spacing )
```

5.54.1.7 nugget_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.nugget_variogram (
              reduced_distance,
              variance_contribution )
```

5.54.1.8 spherical_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.spherical_variogram (
              reduced_distance,
              variance_contribution )
```

5.54.1.9 vectorized_exponential_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_exponential_variogram (
              distance,
              vario_range,
              variance_contribution )
```

Compute the value of a variogram with an exponential model.

Parameters

distance	The distance between the two points
vario_range	The variogram range
variance_contribution	The variance for this variogram model

Returns

The value of the variogram

5.54.1.10 vectorized_gaussian_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_gaussian_variogram (
              distance,
              vario_range,
              variance_contribution )
```

Vectorized theoretical variogram.

Compute the value of a variogram with a Gaussian model

```
@param distance: The distance between the two points
@param vario_range: The variogram range
@param variance_contribution: The variance for this variogram model

@return The value of the variogram
```

5.54.1.11 vectorized_spherical_variogram()

```
def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_spherical_variogram (
              distance,
              vario_range,
              variance_contribution )
```

Compute the value of a variogram with a spherical model.

Parameters

distance	The distance between the two points
vario_range	The variogram range
variance_contribution	The variance for this variogram model

Returns

The value of the variogram

5.54.2 Variable Documentation

5.54.2.1 nogil

```
pyinsar.processing.machine_learning.geostatistics.variogram.nogil
```

5.54.2.2 nopython

```
pyinsar.processing.machine_learning.geostatistics.variogram.nopython
```

2D experimental variogram

Compute the variogram values for a range of distances.

Utilities.

Compute the value of a (possibly nested) 2D variogram.

Compute the value of a variogram with an exponential model.

Compute the value of a variogram with a spherical model.

Compute the value of a variogram with a Gaussian model.

Compute the value of a variogram with a pure nugget effect.

Compute the isotropic experimental variogram on a regular grid

```
@param value_array: A 2D NumPy array with the variable to study
@param grid_yx_spacing: The cell size along each axis (y, x)
@param number_of_lags: Number of lags to compute the variogram
@param lag_unit_distance: Distance between each lag
@param tolerance: Tolerance to add a pair of points to a given lag (if None,
                  set to 0.6*lag_unit_distance)
@param sampling: The proportion of cells of the array to take into account
@param no_data_value: The no-data value

@return A 2D Numpy array containing for each lag the variogram values before
        dividing by the number of pairs of points, the number of pairs of
        points, and the lag values
```

Parameters

reduced_distance	The distance between the two points divided by the variogram range
variance_contribution	The variance for this variogram model

Returns

The value of the variogram

Parameters

delta_y	The distance between the two points along the y axis
delta_x	The distance between the two points along the x axis
vario_models	The models for the variogram
vario_sills	The sills for the variogram
vario_ranges	The major and minor ranges for the variogram
rotation_matrix	The 2D rotation matrix

Returns

The value of the variogram

Map the variogram values in a 2D neighborhood

```
@param vario_models: The models for the variogram
@param vario_sills: The sills for the variogram
@param vario_azimuth: The azimuth of the variogram's major axis (in degree)
@param vario_ranges: The major and minor ranges for the variogram
@param neighborhood_range: The range of the neighborhood, beyond which the
                           cells with already a value are not taken into
                           account (in grid spacing unit)
@param map_shape: The number of cells along each axis (y, x) for the variogram map
@param grid_spacing: The cell size along each axis (y, x)

@return The variogram map as a 2D array
```

Parameters

deltas_y	A 1D array of distances between the two points along the y axis
deltas_x	A 1D array of distances between the two points along the x axis
vario_models	The models for the variogram
vario_sills	The sills for the variogram
vario_azimuth	The azimuth of the variogram's major axis (in degree)
vario_ranges	The major and minor ranges for the variogram

Returns

The variogram values as a 1D array

5.54.2.3 True

```
pyinsar.processing.machine_learning.geostatistics.variogram.True
```

5.55 pyinsar.processing.utilities Namespace Reference

Namespaces

- ann
- deformations
- generic
- insar_simulator_utils
- machine_learning

5.56 pyinsar.processing.utilities.ann Namespace Reference

Functions

- def buildCNN (image_height, image_width, model_dir, rate=0.01, config=None)

  Build a convolutional neural network.
- def train (image_data, image_labels, model_dir, batch_size, num_epochs, max_batches=None, status_line_↩ rate=50, target='', shuffle=True, config=None)

  Train neural network.
- def classify (image_data, model_dir, batch_size=2000, config=None)

  Classify data.
- def length_after_valid_window (length, window, stride)

  Length of dimension after convolving using the padding type 'valid' or using max pooling.
- def shuffleTrainingData (data, labels)

  Shuffles data.
- def restoreGraph (model_dir)

  Restore a network.

5.56.1 Function Documentation

5.56.1.1 buildCNN()

```
def pyinsar.processing.utilities.ann.buildCNN (
              image_height,
              image_width,
              model_dir,
              rate = 0.01,
              config = None )
```

Build a convolutional neural network.

Parameters

image_height	Height of image in pixels
image_width	Width of image in pixels
model_dir	Directory to save network too
rate	Learning rate
config	Config to pass to tf.Session

5.56.1.2 classify()

```
def pyinsar.processing.utilities.ann.classify (
              image_data,
              model_dir,
              batch_size = 2000,
              config = None )
```

Classify data.

Parameters

image_data	Input data
model_dir	Directory where network is stored
batch_size	Batch size to use for classifying data
config	Config to pass on to tf.Session

Returns

Predicted labels for input data

5.56.1.3 length_after_valid_window()

```
def pyinsar.processing.utilities.ann.length_after_valid_window (
              length,
              window,
              stride )
```

Length of dimension after convolving using the padding type 'valid' or using max pooling.

Parameters

length	Initial length
window	Size of convolution window
stride	Stride used

Returns

New size after using convolution with 'valid' padding type or from max pooling

5.56.1.4 restoreGraph()

```
def pyinsar.processing.utilities.ann.restoreGraph (
              model_dir )
```

Restore a network.

Parameters

model_dir	Directory containing network

Returns

graph, operation dictionary, and checkpoint

5.56.1.5 shuffleTrainingData()

```
def pyinsar.processing.utilities.ann.shuffleTrainingData (
              data,
              labels )
```

Shuffles data.

Parameters

data	Input data
labels	Input labels

5.56.1.6 train()

```
def pyinsar.processing.utilities.ann.train (
              image_data,
              image_labels,
              model_dir,
              batch_size,
              num_epochs,
              max_batches = None,
              status_line_rate = 50,
              target = '',
              shuffle = True,
              config = None )
```

Train neural network.

Parameters

image_data	Image data to train (shape [:,image_width, image_height])
image_labels	Corresponding labels
model_dir	Directory where network is stored
batch_size	Batch size
num_epochs	Number of epochs
max_batches	Max number of patches (Typically used for testing)
status_line_rate	Number of batches between outputting training information
target	Unused
shuffle	Whether or not to shuffle the training data
config	Config to pass to tf.Session

5.57 pyinsar.processing.utilities.deformations Namespace Reference

Functions

- def calc_bounding_box (image)

  Calculate bounding box of an object in an image.
- def determine_deformation_bounding_box (deformations)

  Determine bounds around a deformation.
- def determine_x_y_bounds (deformations, x_array, y_array, offset=5000)

  Determine the x and y positions that bound a deformation.

5.57.1 Function Documentation

5.57.1.1 calc_bounding_box()

```
def pyinsar.processing.utilities.deformations.calc_bounding_box (
              image )
```

Calculate bounding box of an object in an image.

Parameters

image	Input image

Returns

Extent of deformation in image (x_start, x_end, y_start, y_end)

5.57.1.2 determine_deformation_bounding_box()

```
def pyinsar.processing.utilities.deformations.determine_deformation_bounding_box (
              deformations )
```

Determine bounds around a deformation.

Parameters

deformations	Input deformations

Returns

Bounding box large enough to include deformation in all directions (x_start, x_end, y_start, y_end)

5.57.1.3 determine_x_y_bounds()

```
def pyinsar.processing.utilities.deformations.determine_x_y_bounds (
              deformations,
              x_array,
              y_array,
              offset = 5000 )
```

Determine the x and y positions that bound a deformation.

Parameters

deformations	Input deformations
x_array	X coordinates
y_array	Y coordinates
offset	Extra padding around measured bounds

Returns

Bounds in units of x_array and y_array with padding (x_start, x_end, y_start, y_end)

5.58 pyinsar.processing.utilities.generic Namespace Reference

Classes

- class FindNearestPixel

    Find the nearest given a time.
- class OrbitInterpolation

    Class for interpolating satellite positions.

Functions

- def get_image_extents (geotransform, shape)

    Get extents of in projection coordinates.
- def proj4StringToDictionary (proj4_string)

    Convert a proj4 string into a dictionary.
- def sorted_alphanumeric (l)

    Sort a list of strings with numbers.
- def phase_shift (data, phase)

    Apply a phase shift to data.
- def find_closest_time (time, date)

    Find the closest time to a date.
- def rotate (col_vectors, az, ay, ax, dtype=np.float64)

    Rotate 3 dimensional column vectors.
- def translate (col_vectors, delta_x, delta_y, delta_z)

    Translate 3 dimensional column vectors.
- def coherence (s1, s2, window, topo_phase=0)

    This function computes the coherence between two SLCs.
- def scale_image (input_data, vmin=None, vmax=None)
- def keypoints_align (img1, img2, max_matches=40, invert=True)

    **** In Development *** Determine transformation matrix for aligning images*
- def subarray_slice (index, num_items)

    Returns a slice that selects for selecting a chunk out of an array.
- def find_data_asf (lat, lon, processingLevel='SLC', platform='Sentinel-1A, Sentinel, B, kwargs)

    Search Alaska Satellite Facility for data.
- def select_max_matched_data (sentinel_data_list)

    Select the data that can be combined into an interferogram.
- def match_data (sentinel_data_list)

    Seperate into sets of overlapping data.
- def find_earthquake_pairs (organized_data, date)

    Select image pairs around a specified date.
- def generateMatplotlibRectangle (extent, kwargs)

    Generate a matplotlib rectangle from a extents.
- def project_insar_data (in_dataset, lon_center, lat_center, interpolation=gdal.GRA_Cubic, no_data_value=np.↩ nan, data_type=gdal.GDT_Float64)

    Project InSAR data using GDAL.

5.58.1 Function Documentation

5.58.1.1 coherence()

```
def pyinsar.processing.utilities.generic.coherence (
              s1,
              s2,
              window,
              topo_phase = 0 )
```

This function computes the coherence between two SLCs.

The coherence is estimated using an equation presented in InSAR a practical approach, equation 2.7

Parameters

s1	The first single look complex image
s2	The second single look complex image
window	Tuple specifing y, and x window size
topo_phase	Change in phase due to topography

Returns

Numpy array of the coherence

5.58.1.2 find_closest_time()

```
def pyinsar.processing.utilities.generic.find_closest_time (
              time,
              date )
```

Find the closest time to a date.

Parameters

time	Pandas series of datetimes
date	Input date

Returns

Index of closest time to date

5.58.1.3 find_data_asf()

```
def pyinsar.processing.utilities.generic.find_data_asf (
              lat,
              lon,
              processingLevel = 'SLC',
              platform = 'Sentinel-1A,
              Sentinel,
              B,
              kwargs )
```

Search Alaska Satellite Facility for data.

Parameters

lat	Latitude
lon	Longitude
processingLevel	Processing level of data
platform	Instrument to search
kwargs	All additional kwargs will be used to search ASF See https://www.asf.alaska.edu/get-data/learn-by-doing/

Returns

: List of available data matching the search criteria

5.58.1.4 find_earthquake_pairs()

```
def pyinsar.processing.utilities.generic.find_earthquake_pairs (
              organized_data,
              date )
```

Select image pairs around a specified date.

Parameters

organized_data	Dictionary of information about data that has been organized into overlapping images
date	Date of the event of interest

Returns

Dictionary containing lists of pairs of images around the specified event

5.58.1.5 generateMatplotlibRectangle()

```
def pyinsar.processing.utilities.generic.generateMatplotlibRectangle (
              extent,
              kwargs )
```

Generate a matplotlib rectangle from a extents.

Parameters

extent	Container holding the extent (x_min, x_max, y_min, y_max)
kwargs	Extra keyword arguments passed to matplotlib.patches.Rectangle

Returns

Matplotlib rectangle

5.58.1.6 get_image_extents()

```
def pyinsar.processing.utilities.generic.get_image_extents (
              geotransform,
              shape )
```

Get extents of in projection coordinates.

Parameters

geotransform	Geo transform for converting between pixel and projected coordinates
shape	Shape of image

5.58.1.7 keypoints_align()

```
def pyinsar.processing.utilities.generic.keypoints_align (
              img1,
              img2,
              max_matches = 40,
              invert = True )
```

$***$ In Development $***$ Determine transformation matrix for aligning images

Parameters

img1	First image
img2	Second image
max_matches	Maximum number of matches between the two images
invert	Invert the transformation matrix

Returns

: Transformation matrix that connects two images

5.58.1.8 match_data()

```
def pyinsar.processing.utilities.generic.match_data (
              sentinel_data_list )
```

Seperate into sets of overlapping data.

Seperates based on relative orbit, track, and frame

Parameters

sentinel_data_list	List of information for different images

Returns

: Dictionary of lists of overlapping data

5.58.1.9 phase_shift()

```
def pyinsar.processing.utilities.generic.phase_shift (
              data,
              phase )
```

Apply a phase shift to data.

Parameters

data	Input data
phase	Input phase

Returns

data shifted by phase

5.58.1.10 proj4StringToDictionary()

```
def pyinsar.processing.utilities.generic.proj4StringToDictionary (
              proj4_string )
```

Convert a proj4 string into a dictionary.

Statements with no value are given a value of None

Parameters

proj4_string	Proj4 string

Returns

Dictionary containing proj4 parameters as a OrderedDict

5.58.1.11 project_insar_data()

```
def pyinsar.processing.utilities.generic.project_insar_data (
              in_dataset,
              lon_center,
              lat_center,
              interpolation = gdal.GRA_Cubic,
              no_data_value = np.nan,
              data_type = gdal.GDT_Float64 )
```

Project InSAR data using GDAL.

Parameters

in_dataset	GDAL data set to be projected
lon_center	Longitude center of projecting
lat_center	Latitude center of projecting
interpolation	What kind of interpolation to use (GDAL Flags)
no_data_value	What value to use in the case of no data
data_type	Resulting data type (GDAL flag)

Returns

array containing projected data

5.58.1.12 rotate()

```
def pyinsar.processing.utilities.generic.rotate (
              col_vectors,
              az,
              ay,
              ax,
              dtype = np.float64 )
```

Rotate 3 dimensional column vectors.

Parameters

col_vectors	Array of column vectors
az	Angle for rotation about the z axis
ay	Angle for rotation about the y axis
ax	Angle for rotation about the x axis
dtype	Data type to use
return	Rotated vectors

5.58.1.13 scale_image()

```
def pyinsar.processing.utilities.generic.scale_image (
              input_data,
              vmin = None,
              vmax = None )
```

5.58.1.14 select_max_matched_data()

```
def pyinsar.processing.utilities.generic.select_max_matched_data (
              sentinel_data_list )
```

Select the data that can be combined into an interferogram.

The particular frame and track that maximizes the number of useable data is chosen

Parameters

sentinel_data_list	

Returns

:

5.58.1.15 sorted_alphanumeric()

```
def pyinsar.processing.utilities.generic.sorted_alphanumeric (
              l )
```

Sort a list of strings with numbers.

Parameters

l	The list

Returns

The sorted list

5.58.1.16 subarray_slice()

```
def pyinsar.processing.utilities.generic.subarray_slice (
              index,
              num_items )
```

Returns a slice that selects for selecting a chunk out of an array.

Parameters

index	Which chunk to select
num_items	Number of items in a chunk

Returns

A slice for selecting index∗num_items to (index+1)∗num_items

5.58.1.17 translate()

```
def pyinsar.processing.utilities.generic.translate (
              col_vectors,
              delta_x,
              delta_y,
              delta_z )
```

Translate 3 dimensional column vectors.

Parameters

col_vectors	Array of column vectors
delta_x	Move this many units in the x direction
delta_y	Move this many units in the y direction
delta_z	Move this many units in the z direction

Returns

Translated vectors

5.59 pyinsar.processing.utilities.insar_simulator_utils Namespace Reference

Functions

- def wrap (x, to_2pi=False)

  Wrap a float or an array.
- def crop_array_from_center (array, crop_shape)

  Crop an array along its borders.
- def mask_deformation (deformation, threshold_function=threshold_li)

  Mask image using a threshold function.
- def calc_bounding_box (image, threshold_function=threshold_li)

  Calcluate the bounding box around an image using the li threshold.
- def retrieve_bounds (thresh_image)

  Retrieve the bounds of an image that has been thesholded.
- def crop_nans (image)

  Shrink image by removing nans.
- def determine_deformation_bounding_box (deformations, largest_box=True, kwargs)

  Calculate the extent of the deformation in image coordinates.
- def determine_x_y_bounds (deformations, x_array, y_array, offset=5000, kwargs)

  Determine the x and y coordinates of the extent of the deformation.
- def generate_interferogram_from_deformation (track_angle, min_ground_range_1, height_1, is_right_looking, wavelength, k, deformation, xx, yy, projected_topography=None, min_ground_range_2=None, height_2=None)

  Generate an interferogram from deformations.
- def old_generate_interferogram_from_deformation (track_angle, min_ground_range, height, is_right_looking, wavelength, k, deformation, xx, yy, projected_topography=None)

Generate an interferogram from deformations.

- def change_in_range_to_phase (los_deformation, wavelength, k=2)

  Compute phase from change in range.
- def phase_to_change_in_range (phase, wavelength, k=2)

  Compute change in range from phase.

5.59.1 Function Documentation

5.59.1.1 calc_bounding_box()

```
def pyinsar.processing.utilities.insar_simulator_utils.calc_bounding_box (
              image,
              threshold_function = threshold_li )
```

Calcluate the bounding box around an image using the li threshold.

Parameters

image	Input image
threshold_function	Threshold function to use

Returns

Extents of a bounding box around the contents in the image (x_min, x_max, y_min, y_max)

5.59.1.2 change_in_range_to_phase()

```
def pyinsar.processing.utilities.insar_simulator_utils.change_in_range_to_phase (
              los_deformation,
              wavelength,
              k = 2 )
```

Compute phase from change in range.

Parameters

los_deformation	Change in distance along line of site
wavelength	Wavelength of radar
k	Number of passes

Returns

phase due to change in

5.59.1.3 crop_array_from_center()

```
def pyinsar.processing.utilities.insar_simulator_utils.crop_array_from_center (
              array,
              crop_shape )
```

Crop an array along its borders.

Parameters

array	The array
crop_shape	The number of cells to remove along the y and x axes

Returns

The cropped array

5.59.1.4 crop_nans()

```
def pyinsar.processing.utilities.insar_simulator_utils.crop_nans (
              image )
```

Shrink image by removing nans.

Parameters

image	Input image

Returns

: Image cropped around valid data

5.59.1.5 determine_deformation_bounding_box()

```
def pyinsar.processing.utilities.insar_simulator_utils.determine_deformation_bounding_box (
              deformations,
              largest_box = True,
              kwargs )
```

Calculate the extent of the deformation in image coordinates.

Parameters

deformations	Input deformations
largest_box	Choose a bounding max that encomposses all selected values in all dimensions
kwargs	Any additional keyword arguments passed to calc_bounding_box

Returns

Extents deformations (x_min, x_max, y_min, y_max)

5.59.1.6 determine_x_y_bounds()

```
def pyinsar.processing.utilities.insar_simulator_utils.determine_x_y_bounds (
              deformations,
              x_array,
              y_array,
              offset = 5000,
              kwargs )
```

Determine the x and y coordinates of the extent of the deformation.

Parameters

deformations	Input deformations
x_array	x coordinates
y_array	y coordinatse
offset	Size to extend the extents of the box
kwargs	Any additional keyword arguments passed to determine_deformation_bounding_box

Returns

Extents of the deformation plus the offset (x_min, x_max, y_min, y_max)

5.59.1.7 generate_interferogram_from_deformation()

```
def pyinsar.processing.utilities.insar_simulator_utils.generate_interferogram_from_deformation (
              track_angle,
              min_ground_range_1,
              height_1,
              is_right_looking,
              wavelength,
              k,
              deformation,
              xx,
              yy,
              projected_topography = None,
              min_ground_range_2 = None,
              height_2 = None )
```

Generate an interferogram from deformations.

Parameters

track_angle	Satellite track angle
min_ground_range↩ _1	Minimum ground range to deformations for first pass
height_1	Height of satellite for first pass
is_right_looking	The satellite is looking to the right
wavelength	Wavelength of the signal
k	number of passes (1 or 2)
deformation	map of deformation
xx	x coordinates of deformation
yy	y coordinates of deformation
projected_topography	Elevation data
min_ground_range↩ _2	Minimum ground range to deformations for second pass
height_2	Height of satellite for second pass

Returns

Inteferogram due to the deformations

5.59.1.8 mask_deformation()

```
def pyinsar.processing.utilities.insar_simulator_utils.mask_deformation (
              deformation,
              threshold_function = threshold_li )
```

Mask image using a threshold function.

Parameters

deformation	Deformation to mask
threshold_function	Function to calculate the threshold value

Returns

Masked image

5.59.1.9 old_generate_interferogram_from_deformation()

```
def pyinsar.processing.utilities.insar_simulator_utils.old_generate_interferogram_from_deformation
(
              track_angle,
              min_ground_range,
              height,
              is_right_looking,
              wavelength,
              k,
              deformation,
              xx,
              yy,
              projected_topography = None )
```

Generate an interferogram from deformations.

Parameters

track_angle	Satellite track angle
min_ground_range	Minimum ground range to deformations
height	Height of satellite
is_right_looking	The satellite is looking to the right
wavelength	Wavelength of the signal
k	number of passes (1 or 2)
deformation	map of deformation
xx	x coordinates of deformation
yy	y coordinates of deformation
projected_topography	Elevation data

Returns

Inteferogram due to the deformations

5.59.1.10 phase_to_change_in_range()

```
def pyinsar.processing.utilities.insar_simulator_utils.phase_to_change_in_range (
              phase,
              wavelength,
              k = 2 )
```

Compute change in range from phase.

Parameters

phase	Input phase
wavelength	Wavelength of radar
k	Number of passes

Returns

Change in range

5.59.1.11 retrieve_bounds()

```
def pyinsar.processing.utilities.insar_simulator_utils.retrieve_bounds (
              thresh_image )
```

Retrieve the bounds of an image that has been thesholded.

Parameters

thresh_image	Image filled with ones for valid and zeros for invalid

Returns

: Extents of a rectangle around valid data (x_start, x_end, y_start, y_end)

5.59.1.12 wrap()

```
def pyinsar.processing.utilities.insar_simulator_utils.wrap (
              x,
              to_2pi = False )
```

Wrap a float or an array.

Parameters

x	The float or array
to_2pi	If True, wrap to [0, 2pi) instead of [-pi, pi]

Returns

The wrapped array (in radian between -pi and pi)

5.60 pyinsar.processing.utilities.machine_learning Namespace Reference

Classes

- class DataRetriever

  Class for retrieving data from an hdf file.

Functions

- def divide_into_squares (image, size, stride)

  Create many patches from an image.
- def generate_minimum_ground_range_limits (satellite_height, incidence_ranges, image_size)

  Determine the limits of minimum ground ranges of a satellite pass.
- def generate_phase_samples_from_looks_and_ranges (deformation_list, xx, yy, satellite_height, track_angles, minimum_ground_ranges, size=(100, 100), dtype=np.float32)

  Generates different possible phases from a list of deformations due to different track angles and groud ranges.
- def generate_phase_samples (deformation, satellite_height, radar_wavelength, cell_size, image_size, stride=20)

  ***In Development** Generate phase samples by tiling an array of deformations*
- def rotate_image_list (in_image_extents, in_image_list, progress=True)

  Rotate input images 0, 90, 180, and 270 degrees.

5.60.1 Function Documentation

5.60.1.1 divide_into_squares()

```
def pyinsar.processing.utilities.machine_learning.divide_into_squares (
              image,
              size,
              stride )
```

Create many patches from an image.

Will drop any patches that contain NaN's

Parameters

image	Source image
size	Size of one side of the square patch
stride	Spacing between patches (must be an integer greater than 0)

Returns

List containing the extent of each patch and a list of the patches

5.60.1.2 generate_minimum_ground_range_limits()

```
def pyinsar.processing.utilities.machine_learning.generate_minimum_ground_range_limits (
              satellite_height,
              incidence_ranges,
              image_size )
```

Determine the limits of minimum ground ranges of a satellite pass.

Parameters

satellite_height	Height of satellite
incidence_ranges	Range of valid incidence angles (shape of [:,2])
image_size	Length of image

Returns

range of possible minimum ground ranges

5.60.1.3 generate_phase_samples()

```
def pyinsar.processing.utilities.machine_learning.generate_phase_samples (
              deformation,
              satellite_height,
              radar_wavelength,
              cell_size,
              image_size,
              stride = 20 )
```

In Development Generate phase samples by tiling an array of deformations

Parameters

deformation	Array containing deformation
satellite_height	Height of Satellite
radar_wavelength	Wavelength of radar
cell_size	Size of cell (length of one side of the cell)
image_size	Ignored?
stride	Distance between tiles

5.60.1.4 generate_phase_samples_from_looks_and_ranges()

```
def pyinsar.processing.utilities.machine_learning.generate_phase_samples_from_looks_and_ranges (
              deformation_list,
              xx,
              yy,
              satellite_height,
              track_angles,
              minimum_ground_ranges,
              size = (100,100),
              dtype = np.float32 )
```

Generates different possible phases from a list of deformations due to different track angles and groud ranges.

Parameters

deformation_list	List of deformations
xx	x coordinates
yy	y coordinates
satellite_height	Height of satellite
track_angles	Iterable of track angles
minimum_ground_ranges	Iterable of minimum ground ranges
size	Tuple giving the size of each deformation in deformation_list
dtype	Data type

Returns

array containing paramaters and and array containing phases

5.60.1.5 rotate_image_list()

```
def pyinsar.processing.utilities.machine_learning.rotate_image_list (
              in_image_extents,
              in_image_list,
              progress = True )
```

Rotate input images 0, 90, 180, and 270 degrees.

Parameters

in_image_extents	List of the extents of the images being rotated
in_image_list	List of images to rotate
progress	Show a progress bar

Returns

array of image extents, and array of rotated images

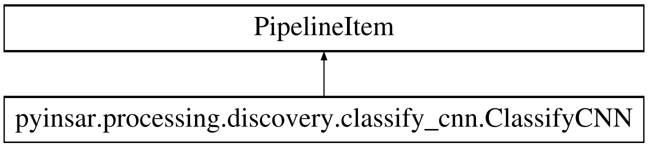
Chapter 6

Class Documentation

6.1 pyinsar.processing.discovery.ClassifyCNN Class Reference

Train a CNN.

Inheritance diagram for pyinsar.processing.discovery.ClassifyCNN:



Public Member Functions

- def __init__ (self, str_description, cnn_network_dir, batch_size=2000, config=None, compare_labels=False, stride=None, size=None)

  Initialize TrainCNN item.
- def process (self, obj_data)

  Classify data using a CNN using data in Image wrapper.

Public Attributes

- cnn_network_dir
- batch_size
- config
- compare_labels
- stride
- size

6.1.1 Detailed Description

Train a CNN.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 __init__()

```
def pyinsar.processing.discovery.ClassifyCNN.__init__ (
              self,
              str_description,
              cnn_network_dir,
              batch_size = 2000,
              config = None,
              compare_labels = False,
              stride = None,
              size = None )
```

Initialize TrainCNN item.

Parameters

str_description	String describing item
cnn_network_dir	Strining containing the directiory where the CNN is stored
batch_size	Batch size to use when classifying with Tensorflow
config	Additional session configuration dictionary
compare_labels	Compare measured labels with labels stored in metadata
stride	Distance between images if it necessary to cut image into tiles
size	Size of images to feed into CNN

6.1.3 Member Function Documentation

6.1.3.1 process()

```
def pyinsar.processing.discovery.ClassifyCNN.process (
              self,
              obj_data )
```

Classify data using a CNN using data in Image wrapper.

Parameters

obj_data	Image wrapper

6.1.4 Member Data Documentation

6.1.4.1 batch_size

```
pyinsar.processing.discovery.ClassifyCNN.batch_size
```

6.1.4.2 cnn_network_dir

```
pyinsar.processing.discovery.ClassifyCNN.cnn_network_dir
```

6.1.4.3 compare_labels

```
pyinsar.processing.discovery.ClassifyCNN.compare_labels
```

6.1.4.4 config

```
pyinsar.processing.discovery.ClassifyCNN.config
```

6.1.4.5 size

```
pyinsar.processing.discovery.ClassifyCNN.size
```

6.1.4.6 stride

```
pyinsar.processing.discovery.ClassifyCNN.stride
```

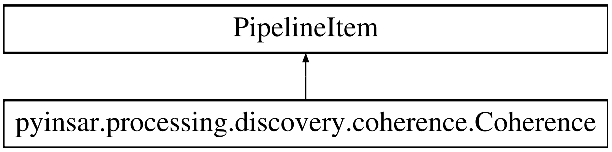
The documentation for this class was generated from the following file:

- processing/discovery/classify_cnn.py

6.2 pyinsar.processing.discovery.Coherence Class Reference

Calculate coherence between single-look complex SAR images.

Inheritance diagram for pyinsar.processing.discovery.Coherence:



Public Member Functions

- def __init__ (self, str_description, window, pairing='neighbor', use_progress_bar=False)

  Initialize coherence pipeline item.
- def process (self, obj_data)

  Compute the coherency between two.

Public Attributes

- window
- pairing
- use_progress_bar

6.2.1 Detailed Description

Calculate coherence between single-look complex SAR images.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 __init__()

```
def pyinsar.processing.discovery.Coherence.__init__ (
              self,
              str_description,
              window,
              pairing = 'neighbor',
              use_progress_bar = False )
```

Initialize coherence pipeline item.

Parameters

str_description	Short string describing item
window	Tuple indicating the y and x window size
pairing	How to pair slc images. "neighbor" computes coherence between neighboring images
use_progress_bar	Display progress using a progress bar

6.2.3 Member Function Documentation

6.2.3.1 process()

```
def pyinsar.processing.discovery.Coherence.process (
              self,
              obj_data )
```

Compute the coherency between two.

Parameters

obj_data	Data wrapper

6.2.4 Member Data Documentation

6.2.4.1 pairing

```
pyinsar.processing.discovery.Coherence.pairing
```

6.2.4.2 use_progress_bar

```
pyinsar.processing.discovery.Coherence.use_progress_bar
```

6.2.4.3 window

```
pyinsar.processing.discovery.Coherence.window
```

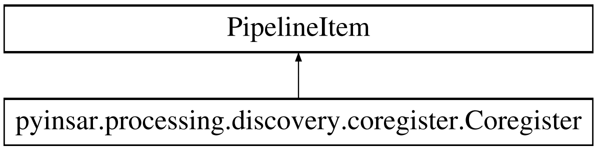
The documentation for this class was generated from the following file:

- processing/discovery/coherence.py

6.3 pyinsar.processing.discovery.Coregister Class Reference

*** In Devolopment *** Pipeline item to coregister images

Inheritance diagram for pyinsar.processing.discovery.Coregister:



Public Member Functions

- def __init__ (self, str_description, ap_paramList, image_limits=None, num_iterations=3)

  Initialize Coregister pipeline item.
- def process (self, obj_data)

  Coregister images.

6.3.1 Detailed Description

*** In Devolopment *** Pipeline item to coregister images

6.3.2 Constructor & Destructor Documentation

6.3.2.1 __init__()

```
def pyinsar.processing.discovery.Coregister.__init__ (
              self,
              str_description,
              ap_paramList,
              image_limits = None,
              num_iterations = 3 )
```

Initialize Coregister pipeline item.

Parameters

str_description	String describing item
ap_paramList[reg_type]	Registration method (currently supports 'imreg_translation', imreg_affine' and 'keypoints'
image_limits	Limits of image to use when comparing for coregistration
num_iterations	Number of iterations (Only used with 'imreg_translation')

6.3.3 Member Function Documentation

6.3.3.1 process()

```
def pyinsar.processing.discovery.Coregister.process (
              self,
              obj_data )
```

Coregister images.

Parameters

obj_data	Image data wrapper

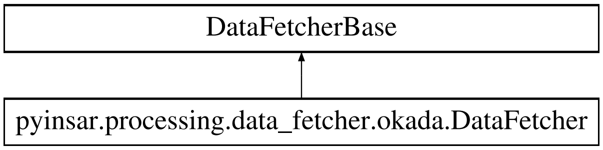
The documentation for this class was generated from the following file:

- processing/discovery/coregister.py

6.4 pyinsar.processing.data_fetcher.okada.DataFetcher Class Reference

Generates data from an Okada model.

Inheritance diagram for pyinsar.processing.data_fetcher.okada.DataFetcher:



Public Member Functions

- def __init__ (self, ap_paramList, xx_array, yy_array, verbose=False)

  Initialize Okada DataFetcher.
- def output (self)

  Output deformation in an image wrapper.

6.4.1 Detailed Description

Generates data from an Okada model.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 __init__()

```
def pyinsar.processing.data_fetcher.okada.DataFetcher.__init__ (
              self,
              ap_paramList,
              xx_array,
              yy_array,
              verbose = False )
```

Initialize Okada DataFetcher.

Parameters

ap_paramList[fault_centroid_x]	x centroid
ap_paramList[fault_centroid_y]	y centroid
ap_paramList[fault_centroid_depth]	Fault depth
ap_paramList[fault_strike]	Fault strike
ap_paramList[fault_dip]	Fault dip
ap_paramList[fault_length]	Fault Length
ap_paramList[fault_width]	Fault width
ap_paramList[fault_rake]	Fault rake
ap_paramList[fault_slip]	Fault slip
ap_paramList[fault_open]	Fault open
ap_paramList[poisson_ratio]	Poisson ratio
xx_array	Array of x coordinates
yy_array	Array of y coordinates
verbose	Print out extra information

6.4.3 Member Function Documentation

6.4.3.1 output()

```
def pyinsar.processing.data_fetcher.okada.DataFetcher.output (
              self )
```

Output deformation in an image wrapper.

Returns

Deformation in an Image wrapper

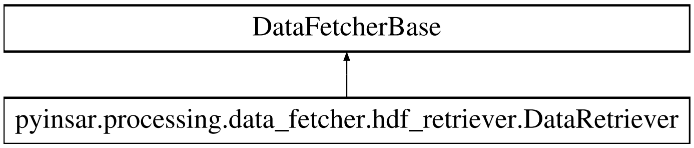
The documentation for this class was generated from the following file:

- processing/data_fetcher/okada.py

6.5 pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever Class Reference

Data fetcher for retrieving hdf image data made for training in convolutional neural networks.

Inheritance diagram for pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever:



Public Member Functions

- def __init__ (self, filename_list, label_list, size, dtype, num_chunks, num_training_items, num_validation_items, num_testing_items)

  Initialize TrainCNN item.
- def perturb (self)
- def output (self)

6.5.1 Detailed Description

Data fetcher for retrieving hdf image data made for training in convolutional neural networks.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 __init__()

```
def pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever.__init__ (
              self,
              filename_list,
              label_list,
              size,
              dtype,
              num_chunks,
              num_training_items,
              num_validation_items,
              num_testing_items )
```

Initialize TrainCNN item.

Parameters

filename_list	List of hdf retriever files
label_list	Label for each file
size	Image shape
dtype	Data type to return
num_chunks	Number of chunks to read in at at time. This is necessary due to a performance issue with h5py
num_training_items	Number of items in each dataset to use for training
num_validation_items	Number of items from each dataset to use for validation
num_testing_items	Number of items in each dataset to use for testing

6.5.3 Member Function Documentation

6.5.3.1 output()

```
def pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever.output (
              self )
```

6.5.3.2 perturb()

```
def pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever.perturb (
              self )
```

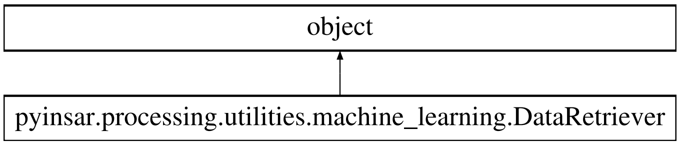
The documentation for this class was generated from the following file:

- processing/data_fetcher/hdf_retriever.py

6.6 pyinsar.processing.utilities.machine_learning.DataRetriever Class Reference

Class for retrieving data from an hdf file.

Inheritance diagram for pyinsar.processing.utilities.machine_learning.DataRetriever:



Public Member Functions

- def __init__ (self, file_name_list, label_list, size, dtype=np.float32, chunk_size=1000)

  Initilaize DataRetriever object.
- def get_num_images (self)

  Get the number of images for each label.
- def get_images (self, index)

  Retrieve images given by index.

Public Attributes

- label_list
- size
- dtype
- chunk_size
- data_file_dict

6.6.1 Detailed Description

Class for retrieving data from an hdf file.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 __init__()

```
def pyinsar.processing.utilities.machine_learning.DataRetriever.__init__ (
              self,
              file_name_list,
              label_list,
              size,
              dtype = np.float32,
              chunk_size = 1000 )
```

Initilaize DataRetriever object.

Parameters

file_name_list	List of hdf filenames
label_list	List of labels
size	Tuple containing the size of the images
dtype	Data type of images
chunk_size	Size of chunks to use when reading data

6.6.3 Member Function Documentation

6.6.3.1 get_images()

```
def pyinsar.processing.utilities.machine_learning.DataRetriever.get_images (
              self,
              index )
```

Retrieve images given by index.

Parameters

index	Array with shape [:,2], first column label, second column index

Returns

Requested images

6.6.3.2 get_num_images()

```
def pyinsar.processing.utilities.machine_learning.DataRetriever.get_num_images (
              self )
```

Get the number of images for each label.

Returns

Number of images associated with each label

6.6.4 Member Data Documentation

6.6.4.1 chunk_size

```
pyinsar.processing.utilities.machine_learning.DataRetriever.chunk_size
```

6.6.4.2 data_file_dict

```
pyinsar.processing.utilities.machine_learning.DataRetriever.data_file_dict
```

6.6.4.3 dtype

```
pyinsar.processing.utilities.machine_learning.DataRetriever.dtype
```

6.6.4.4 label_list

```
pyinsar.processing.utilities.machine_learning.DataRetriever.label_list
```

6.6.4.5 size

```
pyinsar.processing.utilities.machine_learning.DataRetriever.size
```

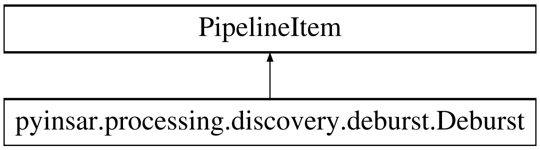
The documentation for this class was generated from the following file:

- processing/utilities/machine_learning.py

6.7 pyinsar.processing.discovery.deburst.Deburst Class Reference

Debursts Sentinel-1 TOPSAR data.

Inheritance diagram for pyinsar.processing.discovery.deburst.Deburst:



Public Member Functions

- def __init__ (self, str_description, cut_on_master=True)

  Initialize Deburst item.
- def process (self, obj_data)

  Preprocesses sentinel 1 data.

6.7.1 Detailed Description

Debursts Sentinel-1 TOPSAR data.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 __init__()

```
def pyinsar.processing.discovery.deburst.Deburst.__init__ (
              self,
              str_description,
              cut_on_master = True )
```

Initialize Deburst item.

Parameters

str_description	String description of item
cut_on_master	Use the master burst cut on slave

6.7.3 Member Function Documentation

6.7.3.1 process()

```
def pyinsar.processing.discovery.deburst.Deburst.process (
              self,
              obj_data )
```

Preprocesses sentinel 1 data.

Parameters

obj_data	Data wrapper

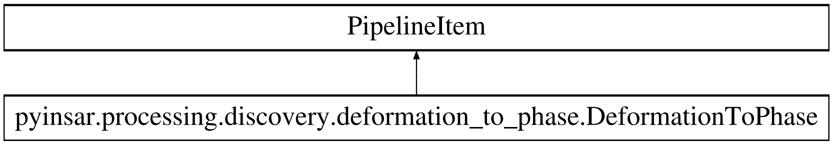
The documentation for this class was generated from the following file:

- processing/discovery/deburst.py

6.8 pyinsar.processing.discovery.DeformationToPhase Class Reference

Convert deformation to phas.

Inheritance diagram for pyinsar.processing.discovery.DeformationToPhase:


PipelineItem
pyinsar.processing.discovery.deformation_to_phase.DeformationToPhase

Public Member Functions

- def __init__ (self, str_description, ap_paramList, xx, yy)

  Initialize Deformation to Phase pipeline item.
- def process (self, obj_data)

  Convert deformations in a data wrapper to phases.

6.8.1 Detailed Description

Convert deformation to phas.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 __init__()

```
def pyinsar.processing.discovery.DeformationToPhase.__init__ (
              self,
              str_description,
              ap_paramList,
              xx,
              yy )
```

Initialize Deformation to Phase pipeline item.

Parameters

str_description	String description of item
ap_paramList[track_angle]	= Auto param of the track angle
ap_paramList[min_ground_range_1]	= Auto param of min_ground_range_1
ap_paramList[height_1]	= Auto param of height_1
ap_paramList[is_right_looking]	= Auto param of is_right_looking (boolean)
ap_paramList[wavelength]	= Auto param of the wavelength for converting deformation to phase
ap_paramList[k]	= Auto param of k
xx	= x coordinates

6.8.3 Member Function Documentation

6.8.3.1 process()

```
def pyinsar.processing.discovery.DeformationToPhase.process (
              self,
              obj_data )
```

Convert deformations in a data wrapper to phases.

Parameters

obj_data	Image data wrapper

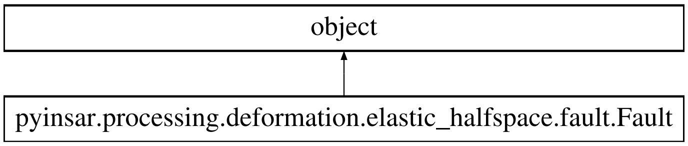
The documentation for this class was generated from the following file:

- processing/discovery/deformation_to_phase.py

6.9 pyinsar.processing.deformation.elastic_halfspace.fault.Fault Class Reference

∗∗∗ In Development ∗∗∗ Model a fault as a collection of small okada faults

Inheritance diagram for pyinsar.processing.deformation.elastic_halfspace.fault.Fault:


object
pyinsar.processing.deformation.elastic_halfspace.fault.Fault

Public Member Functions

- def __init__ (self, x_center, y_center, depth, length, width, strike, dip, num_elements_length, num_elements_width, poisson_ratio=0.25, dtype=np.float32)

  Initialize Fault object.

- def generateDeformation (self, slip, rake, x_coords, y_coords, simple=True)

  Generate surface deformations from fault.

Public Attributes

- x_center
- y_center
- depth
- length
- width
- strike
- dip
- poisson_ratio
- cell_width
- cell_length
- unrotated_x
- unrotated_y
- cell_centroids

6.9.1 Detailed Description

∗∗∗ In Development ∗∗∗ Model a fault as a collection of small okada faults

6.9.2 Constructor & Destructor Documentation

6.9.2.1 __init__()

```
def pyinsar.processing.deformation.elastic_halfspace.fault.Fault.__init__ (
              self,
              x_center,
              y_center,
              depth,
              length,
              width,
              strike,
              dip,
              num_elements_length,
              num_elements_width,
              poisson_ratio = 0.25,
              dtype = np.float32 )
```

Initialize Fault object.

Parameters

x_center	x centroid of fault
y_center	y centroid of fault
depth	Depth to centroid of fault

Parameters

length	Length of fault (along strike)
width	Width of fault (along dip)
strike	Angle from north of the fault direction
dip	Dip angle
num_elements_length	Number of elements in the length direction
num_elements_width	Number of elements in the widht direction
poisson_ratio	Poisson ratio
dtype	Data type to use in calculations

6.9.3 Member Function Documentation

6.9.3.1 generateDeformation()

```
def pyinsar.processing.deformation.elastic_halfspace.fault.Fault.generateDeformation (
              self,
              slip,
              rake,
              x_coords,
              y_coords,
              simple = True )
```

Generate surface deformations from fault.

Parameters

slip	2d array of slip with size (num_elements_width, num_elements_length)
rake	Scalar Rake value
x_coords	2d array of x coordinates
y_coords	2d array of y coordinates
simple	If multiple slips per cell are given, just apply calculate deformation from a combined slip

Returns

Surface deformations at specificed coordinates

6.9.4 Member Data Documentation

6.9.4.1 cell_centroids

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.cell_centroids
```

6.9.4.2 cell_length

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.cell_length
```

6.9.4.3 cell_width

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.cell_width
```

6.9.4.4 depth

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.depth
```

6.9.4.5 dip

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.dip
```

6.9.4.6 length

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.length
```

6.9.4.7 poisson_ratio

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.poisson_ratio
```

6.9.4.8 strike

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.strike
```

6.9.4.9 unrotated_x

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.unrotated_x
```

6.9.4.10 unrotated_y

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.unrotated_y
```

6.9.4.11 width

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.width
```

6.9.4.12 x_center

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.x_center
```

6.9.4.13 y_center

```
pyinsar.processing.deformation.elastic_halfspace.fault.Fault.y_center
```

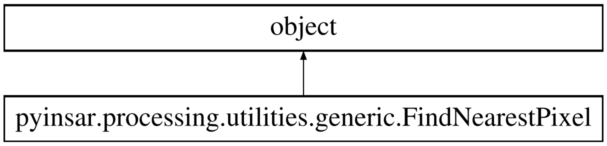
The documentation for this class was generated from the following file:

- processing/deformation/elastic_halfspace/fault.py

6.10 pyinsar.processing.utilities.generic.FindNearestPixel Class Reference

Find the nearest given a time.

Inheritance diagram for pyinsar.processing.utilities.generic.FindNearestPixel:



Public Member Functions

- def __init__ (self, aztime, start_date)

  Initialize FindNearestPixel.
- def __call__ (self, in_time)

  Find the pixel closest to in_time.

6.10.1 Detailed Description

Find the nearest given a time.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 __init__()

```
def pyinsar.processing.utilities.generic.FindNearestPixel.__init__ (
              self,
              aztime,
              start_date )
```

Initialize FindNearestPixel.

Parameters

aztime	Input azimuth time series
start_date	The starting date to use when compting the nearest pixel

6.10.3 Member Function Documentation

6.10.3.1 __call__()

```
def pyinsar.processing.utilities.generic.FindNearestPixel.__call__ (
              self,
              in_time )
```

Find the pixel closest to in_time.

The time is converted to a datetime based on the start_date used to create this object

Parameters

in_time	Input time

Returns

: Pixel that is closest to the input time

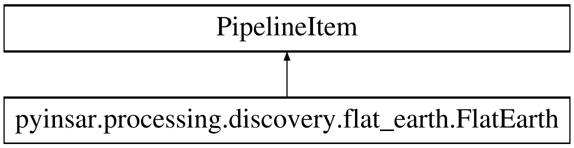
The documentation for this class was generated from the following file:

- processing/utilities/generic.py

6.11 pyinsar.processing.discovery.FlatEarth Class Reference

∗∗∗ In Development ∗∗∗ Remove flat Earth contribution from interferogram

Inheritance diagram for pyinsar.processing.discovery.FlatEarth:



Public Member Functions

- def __init__ (self, str_description, x_range=None, y_range=None, k=2, remove_topography=False, save_↩ correction=False)

    Initialize Flat Earth item.
- def process (self, obj_data)

    Remove flat earth contribution.

Public Attributes

- k

6.11.1 Detailed Description

∗∗∗ In Development ∗∗∗ Remove flat Earth contribution from interferogram

6.11.2 Constructor & Destructor Documentation

6.11.2.1 __init__()

```
def pyinsar.processing.discovery.FlatEarth.__init__ (
              self,
              str_description,
              x_range = None,
              y_range = None,
              k = 2,
              remove_topography = False,
              save_correction = False )
```

Initialize Flat Earth item.

Parameters

str_description	String describing item
x_range	x pixel range to process (None for entire range)
y_range	y pixel range to process (None for entire range)
k	Number of satellite or aircraft passes used to generate the interferogram (1 or 2)
remove_topography	Not implemented
save_correction	Save the image used to correct the interferogram

6.11.3 Member Function Documentation

6.11.3.1 process()

```
def pyinsar.processing.discovery.FlatEarth.process (
              self,
              obj_data )
```

Remove flat earth contribution.

Parameters

obj_data	Input image data wrapper

6.11.4 Member Data Documentation

6.11.4.1 k

```
pyinsar.processing.discovery.FlatEarth.k
```

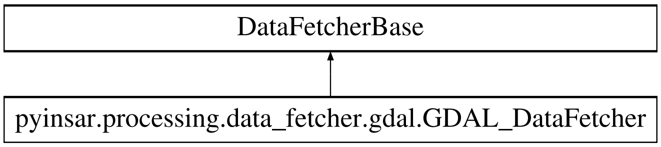
The documentation for this class was generated from the following file:

- processing/discovery/flat_earth.py

6.12 pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher Class Reference

Data fetcher for loading Images produced compatiable with GDAL.

Inheritance diagram for pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher:



Public Member Functions

- def __init__ (self, filename_list, label_list, verbose=False)

  Initialize ISCE data fetcher.
- def output (self)

  Load GDAL data.

6.12.1 Detailed Description

Data fetcher for loading Images produced compatiable with GDAL.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 __init__()

```
def pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher.__init__ (
              self,
              filename_list,
              label_list,
              verbose = False )
```

Initialize ISCE data fetcher.

Parameters

filename_list	List of filenames of ISCE interferograms
label_list	List of strings containing names for the interferograms
verbose	Print extra information

6.12.3 Member Function Documentation

6.12.3.1 output()

```
def pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher.output (
              self )
```

Load GDAL data.

Returns

Image data wrapper

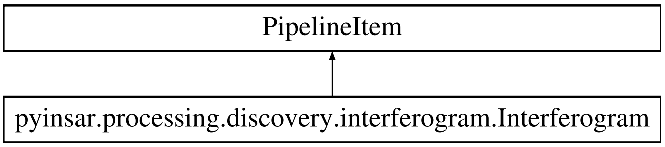
The documentation for this class was generated from the following file:

- processing/data_fetcher/gdal.py

6.13 pyinsar.processing.discovery.interferogram.Interferogram Class Reference

Create Inteferogram from SLC data.

Inheritance diagram for pyinsar.processing.discovery.interferogram.Interferogram:



Public Member Functions

- def __init__ (self, str_description, pairing='neighbor')

  Initialize Interferogram item.
- def process (self, obj_data)

  Create interferograms from SLC images in an image wrapper.

6.13.1 Detailed Description

Create Inteferogram from SLC data.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 __init__()

```
def pyinsar.processing.discovery.interferogram.Interferogram.__init__ (
              self,
              str_description,
              pairing = 'neighbor' )
```

Initialize Interferogram item.

Parameters

str_description	String describing item
pairing	How to pair SLC images. Currently only 'neighbor' is accepted'

6.13.3 Member Function Documentation

6.13.3.1 process()

```
def pyinsar.processing.discovery.interferogram.Interferogram.process (
              self,
              obj_data )
```

Create interferograms from SLC images in an image wrapper.

Parameters

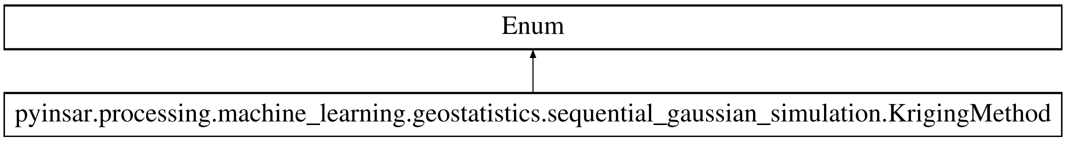
obj_data	Image wrapper containing SLC images

The documentation for this class was generated from the following file:

- processing/discovery/interferogram.py

6.14 pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.↩KrigingMethod Class Reference

Inheritance diagram for pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.Kriging↩Method:


Enum
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod

Static Public Attributes

- int SIMPLE = 0
- int ORDINARY = 1
- nopython

6.14.1 Member Data Documentation

6.14.1.1 nopython

```
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod.↩
nopython  [static]
```

6.14.1.2 ORDINARY

```
int pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.Kriging↩
Method.ORDINARY = 1  [static]
```

6.14.1.3 SIMPLE

```
int pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.Kriging↩
Method.SIMPLE = 0  [static]
```

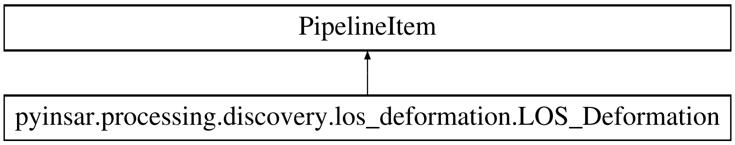
The documentation for this class was generated from the following file:

- processing/machine_learning/geostatistics/sequential_gaussian_simulation.py

6.15 pyinsar.processing.discovery.LOS_Deformation Class Reference

*** In Development ***

Inheritance diagram for pyinsar.processing.discovery.LOS_Deformation:


PipelineItem
pyinsar.processing.discovery.los_deformation.LOS_Deformation

6.15.1 Detailed Description

*** In Development ***

ap_paramList[]

def process(self, obj_data):

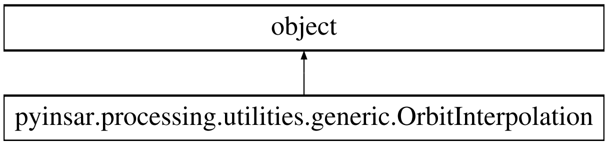
The documentation for this class was generated from the following file:

- processing/discovery/los_deformation.py

6.16 pyinsar.processing.utilities.generic.OrbitInterpolation Class Reference

Class for interpolating satellite positions.

Inheritance diagram for pyinsar.processing.utilities.generic.OrbitInterpolation:



Public Member Functions

- def __init__ (self, orbit_data, time_name='UTC')

  Initilaize orbit interpolation object.
- def get_start_date (self)

  Get starting date used in the interpolation.
- def __call__ (self, in_time, in_datetime=True, interp='position')

  Compute the satellites position or velocity.

6.16.1 Detailed Description

Class for interpolating satellite positions.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 __init__()

```
def pyinsar.processing.utilities.generic.OrbitInterpolation.__init__ (
              self,
              orbit_data,
              time_name = 'UTC' )
```

Initilaize orbit interpolation object.

Parameters

orbit_data	Orbit position data
time_name	Name of time column name in Orbit position data. Set this to None to use the data frame index

6.16.3 Member Function Documentation

6.16.3.1 __call__()

```
def pyinsar.processing.utilities.generic.OrbitInterpolation.__call__ (
              self,
              in_time,
              in_datetime = True,
              interp = 'position' )
```

Compute the satellites position or velocity.

Parameters

in_time	Time of interest
in_datetime	Input is a datetime object (otherwise it's assumed its seconds from start date)
interp	Interpolate "position" or "velocity"

Returns

Satellite position or velocity at in_time

6.16.3.2 get_start_date()

```
def pyinsar.processing.utilities.generic.OrbitInterpolation.get_start_date (
              self )
```

Get starting date used in the interpolation.

Returns

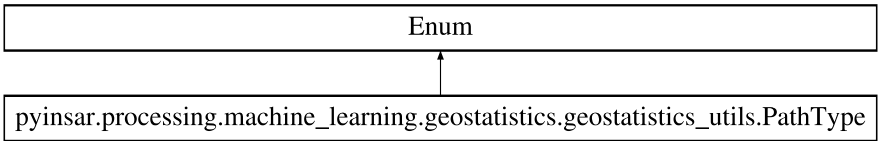
Starting date

The documentation for this class was generated from the following file:

- processing/utilities/generic.py

6.17 pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType Class Reference

Inheritance diagram for pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType:


Enum
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType

Static Public Attributes

- int LINEAR = 0
- int RANDOM = 1

6.17.1 Member Data Documentation

6.17.1.1 LINEAR

```
int pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType.LINEAR = 0
[static]
```

6.17.1.2 RANDOM

```
int pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType.RANDOM = 1
[static]
```

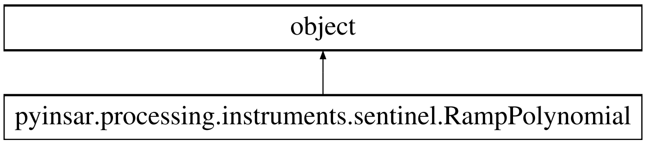
The documentation for this class was generated from the following file:

- processing/machine_learning/geostatistics/geostatistics_utils.py

6.18 pyinsar.processing.instruments.sentinel.RampPolynomial Class Reference

Polynomial used for quantities relating to deramping sentinel.

Inheritance diagram for pyinsar.processing.instruments.sentinel.RampPolynomial:



Public Member Functions

- def __init__ (self, t0, coeff_list, slant_range_time_interval, slant_range_time)

  Initialize Deramp Polynomial object.
- def __call__ (self, t)

  Evaluate the polynomial.

6.18.1 Detailed Description

Polynomial used for quantities relating to deramping sentinel.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 __init__()

```
def pyinsar.processing.instruments.sentinel.RampPolynomial.__init__ (
              self,
              t0,
              coeff_list,
              slant_range_time_interval,
              slant_range_time )
```

Initialize Deramp Polynomial object.

Parameters

t0	Starting time
coeff_list	List of coefficients
slant_range_time_interval	Time between range samples
slant_range_time	Two way slant range time

6.18.3 Member Function Documentation

6.18.3.1 __call__()

```
def pyinsar.processing.instruments.sentinel.RampPolynomial.__call__ (
              self,
              t )
```

Evaluate the polynomial.

Parameters

t	Input time

Returns

Value of polynomial at time t

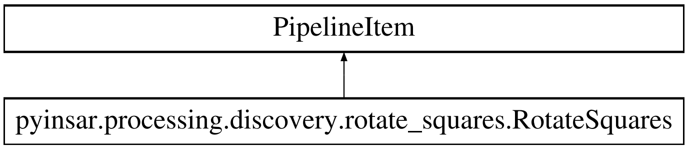
The documentation for this class was generated from the following file:

- processing/instruments/sentinel.py

6.19 pyinsar.processing.discovery.RotateSquares Class Reference

Generate new images by rotating subsections of data defined by Shapely squares.

Inheritance diagram for pyinsar.processing.discovery.RotateSquares:



Public Member Functions

- def __init__ (self, str_description, ap_paramList, square_result_name, angles, clean=True)

  Initialize RotateSquares object.
- def process (self, obj_data)

  Generate rotated images based on Shapely squares.

6.19.1 Detailed Description

Generate new images by rotating subsections of data defined by Shapely squares.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 __init__()

```
def pyinsar.processing.discovery.RotateSquares.__init__ (
              self,
              str_description,
              ap_paramList,
              square_result_name,
              angles,
              clean = True )
```

Initialize RotateSquares object.

Parameters

str_description	String describing class
ap_paramList[SplineOrder]	Spline order used in interpolation
square_result_name	Name of pipeline item that contains the Shapely squares
angles	Angles used when rotating squares
clean	Remove any squares that contain NaN's

6.19.3 Member Function Documentation

6.19.3.1 process()

```
def pyinsar.processing.discovery.RotateSquares.process (
              self,
              obj_data )
```

Generate rotated images based on Shapely squares.

Parameters

obj_data	Image data wrapper

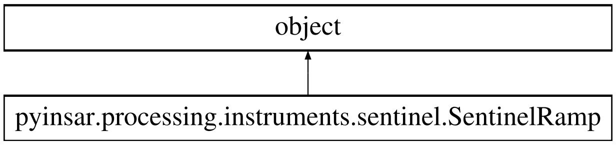
The documentation for this class was generated from the following file:

- processing/discovery/rotate_squares.py

6.20 pyinsar.processing.instruments.sentinel.SentinelRamp Class Reference

Calcuate the combined ramp and modulated phase in Sentinel.

Inheritance diagram for pyinsar.processing.instruments.sentinel.SentinelRamp:



Public Member Functions

- def __init__ (self, metadata, modulation=True)

    Initialize Sentiel Ramp.
- def __call__ (self, lines, samples, index)

    Calculate the phase change from the Sentinel ramp and modulation.

Public Attributes

- modulation

6.20.1 Detailed Description

Calcuate the combined ramp and modulated phase in Sentinel.

This class was created following the guide at: `https://sentinel.esa.int/documents/247904/1653442/Sentinel-1-TOPS-SLC_Deramping`

6.20.2 Constructor & Destructor Documentation

6.20.2.1 __init__()

```
def pyinsar.processing.instruments.sentinel.SentinelRamp.__init__ (
              self,
              metadata,
              modulation = True )
```

Initialize Sentiel Ramp.

Parameters

metadata	ElemenTree containing the SLC metadata
modulation	Whether to include modulation in the ramp

6.20.3 Member Function Documentation

6.20.3.1 __call__()

```
def pyinsar.processing.instruments.sentinel.SentinelRamp.__call__ (
              self,
              lines,
              samples,
              index )
```

Calculate the phase change from the Sentinel ramp and modulation.

Parameters

lines	Index of lines
samples	Index of samples
index	Burst index (starts at 0)

Returns

Phase due to ramp and modulation

6.20.4 Member Data Documentation

6.20.4.1 modulation

```
pyinsar.processing.instruments.sentinel.SentinelRamp.modulation
```

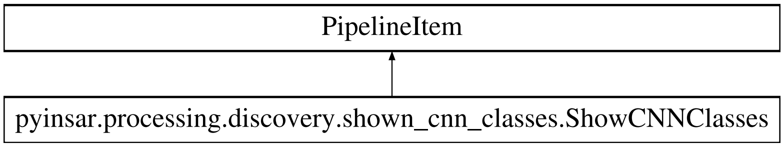
The documentation for this class was generated from the following file:

- processing/instruments/sentinel.py

6.21 pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses Class Reference

Dispay CNN Classifications on segments of an image.

Inheritance diagram for pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses:


PipelineItem
pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses

Public Member Functions

- def __init__ (str_description, class_name, colors)

  Initialize ShowCNNClassesItem.
- def process (self, obj_data)

  Show the images with classifications.

Public Attributes

- class_name
- colors

6.21.1 Detailed Description

Dispay CNN Classifications on segments of an image.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 __init__()

```
def pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses.__init__ (
              str_description,
              class_name,
              colors )
```

Initialize ShowCNNClassesItem.

Parameters

str_description	String name of item
class_name	Name of classes
colors	List of colors containing a color for each class

6.21.3 Member Function Documentation

6.21.3.1 process()

```
def pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses.process (
              self,
              obj_data )
```

Show the images with classifications.

Parameters

obj_data	Image data wrapper

6.21.4 Member Data Documentation

6.21.4.1 class_name

```
pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses.class_name
```

6.21.4.2 colors

```
pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses.colors
```

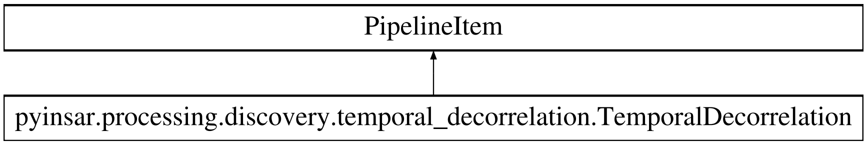
The documentation for this class was generated from the following file:

- processing/discovery/shown_cnn_classes.py

6.22 pyinsar.processing.discovery.TemporalDecorrelation Class Reference

Pipeline item to add temporal decorrelation to some phase.

Inheritance diagram for pyinsar.processing.discovery.TemporalDecorrelation:


PipelineItem
pyinsar.processing.discovery.temporal_decorrelation.TemporalDecorrelation

Public Member Functions

- def __init__ (self, str_description, ap_paramList, grid_yx_spacing, wavelength, seed=None, save_noise=False)

    Initialize Temporal Decorrelation pipeline item.
- def process (self, obj_data)

    Add temporal decorrelation to a phase image.

6.22.1 Detailed Description

Pipeline item to add temporal decorrelation to some phase.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 __init__()

```
def pyinsar.processing.discovery.TemporalDecorrelation.__init__ (
              self,
              str_description,
              ap_paramList,
              grid_yx_spacing,
              wavelength,
              seed = None,
              save_noise = False )
```

Initialize Temporal Decorrelation pipeline item.

Parameters

str_description	String description of item
ap_paramList[vario_models]	= Auto list of SGS models
ap_paramList[vario_sills]	= Auto list of SGS sills
ap_paramList[vario_azimuth]	= Auto param of SGS azimuth
ap_paramList[vario_ranges]	= Auto list of SGS ranges
ap_paramList[max_num_data]	= Auto param of the max size of the neighborhood
ap_paramList[decorrelation_mean]	= Auto param of the decorrelation mean in the same units as the wavelength
ap_paramList[decorrelation_std]	= Auto param of decorrelation standard deviation in the same units as the wavelength
grid_yx_spacing	The y,x grid spacing
wavelength	Wavelength for converting to phase (from path length)
seed	Seed to use when generating noise
save_noise	Boolean indicating whether or not to save a copy of the noise in the results

6.22.3 Member Function Documentation

6.22.3.1 process()

```
def pyinsar.processing.discovery.TemporalDecorrelation.process (
              self,
              obj_data )
```

Add temporal decorrelation to a phase image.

Parameters

obj_data	Image data wrapper

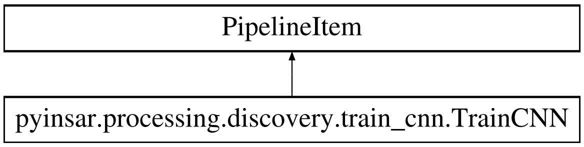
The documentation for this class was generated from the following file:

- processing/discovery/temporal_decorrelation.py

6.23 pyinsar.processing.discovery.TrainCNN Class Reference

Train a CNN.

Inheritance diagram for pyinsar.processing.discovery.TrainCNN:



Public Member Functions

- def __init__ (self, str_description, cnn_network_dir, batch_size, config=None)

  Initialize TrainCNN item.
- def process (self, obj_data)

  Training CNN using data in Image wrapper.

Public Attributes

- cnn_network_dir
- batch_size
- config

6.23.1 Detailed Description

Train a CNN.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 __init__()

```
def pyinsar.processing.discovery.TrainCNN.__init__ (
              self,
              str_description,
              cnn_network_dir,
              batch_size,
              config = None )
```

Initialize TrainCNN item.

Parameters

str_description	String describing item
cnn_network_dir	Strining containing the directiory where the CNN is stored
batch_size	Batch size to use when training data
config	Dictinoary of extra options to use with the tensorflow session

6.23.3 Member Function Documentation

6.23.3.1 process()

```
def pyinsar.processing.discovery.TrainCNN.process (
              self,
              obj_data )
```

Training CNN using data in Image wrapper.

Parameters

obj_data	Image wrapper

6.23.4 Member Data Documentation

6.23.4.1 batch_size

```
pyinsar.processing.discovery.TrainCNN.batch_size
```

6.23.4.2 cnn_network_dir

```
pyinsar.processing.discovery.TrainCNN.cnn_network_dir
```

6.23.4.3 config

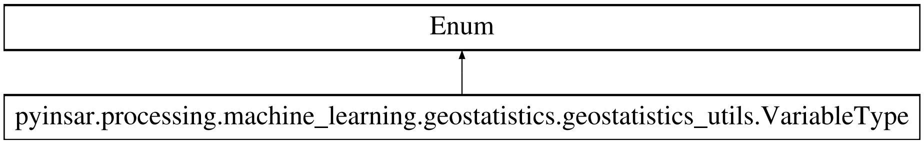
```
pyinsar.processing.discovery.TrainCNN.config
```

The documentation for this class was generated from the following file:

- processing/discovery/train_cnn.py

6.24 pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType Class Reference

Inheritance diagram for pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType:


Enum
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType

Static Public Attributes

- int DISCRETE = 0
- int CONTINUOUS = 1

6.24.1 Member Data Documentation

6.24.1.1 CONTINUOUS

```
int pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType.CONTINUOUS
= 1  [static]
```

6.24.1.2 DISCRETE

```
int pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType.DISCRETE =
0  [static]
```

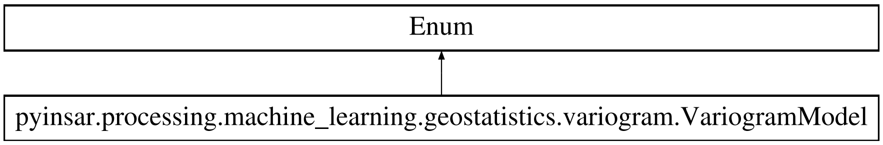
The documentation for this class was generated from the following file:

- processing/machine_learning/geostatistics/geostatistics_utils.py

6.25 pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel Class Reference

2D theoretical variogram

Inheritance diagram for pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel:


Enum
pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel

Static Public Attributes

- int NUGGET = 0
- int GAUSSIAN = 1
- int SPHERICAL = 2
- int EXPONENTIAL = 3

6.25.1 Detailed Description

2D theoretical variogram

6.25.2 Member Data Documentation

6.25.2.1 EXPONENTIAL

```
int pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel.EXPONENTIAL = 3
[static]
```

6.25.2.2 GAUSSIAN

```
int pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel.GAUSSIAN = 1  [static]
```

6.25.2.3 NUGGET

```
int pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel.NUGGET = 0  [static]
```

6.25.2.4 SPHERICAL

```
int pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel.SPHERICAL = 2  [static]
```

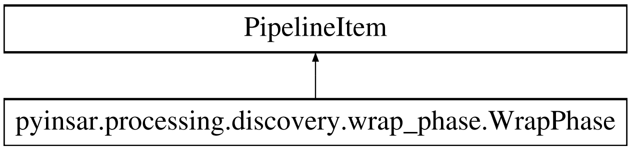
The documentation for this class was generated from the following file:

- processing/machine_learning/geostatistics/variogram.py

6.26 pyinsar.processing.discovery.WrapPhase Class Reference

Pipeline Item that wraps phase.

Inheritance diagram for pyinsar.processing.discovery.WrapPhase:



Public Member Functions

- def process (self, obj_data)
  Wrap phase of images.

6.26.1 Detailed Description

Pipeline Item that wraps phase.

6.26.2 Member Function Documentation

6.26.2.1 process()

```
def pyinsar.processing.discovery.WrapPhase.process (
              self,
              obj_data )
```

Wrap phase of images.

Parameters

obj_data	Image data wrapper

The documentation for this class was generated from the following file:

- processing/discovery/wrap_phase.py

Chapter 7

File Documentation

7.1 data_import/import_georaster.py File Reference

Namespaces

- pyinsar.data_import.import_georaster

Functions

- def pyinsar.data_import.import_georaster.open_georaster (georaster_path, read_only=True)

  Open a georaster with GDAL.
- def pyinsar.data_import.import_georaster.get_georaster_array (gdal_georaster, remove_ndv=True, as_↩ float=True)

  Get a NumPy array from a georaster opened with GDAL.
- def pyinsar.data_import.import_georaster.get_georaster_extent (gdal_georaster)

  Get the extent of a georaster opened with GDAL.
- def pyinsar.data_import.import_georaster.print_georaster_info (gdal_georaster)

  Print some information about the GDAL georaster.

7.2 data_import/import_raster.py File Reference

Namespaces

- pyinsar.data_import.import_raster

Functions

- def pyinsar.data_import.import_raster.read_rsc_header_file (file_path)

  Import GACOS runs.
- def pyinsar.data_import.import_raster.open_gacos_tropospheric_delays (tropodelay_header_path)

  Open a topospheric delay map computed by the Generic Atmospheric Correction Online Service for InSAR (GACOS)
- def pyinsar.data_import.import_raster.open_sgems_file (file_location)

  Import SGEMS files.
- def pyinsar.data_import.import_raster.open_sgems_file_from_url (file_url)

  Open an SGEMS file containing one or several variables in an array from the file's URL.

7.3 data_import/import_srcmod.py File Reference

Namespaces

- pyinsar.data_import.import_srcmod

Functions

- def pyinsar.data_import.import_srcmod.read_srcmod_data (srcmod_data, dtype=np.float64, skip_sanity_↩ check=False)

  ∗∗∗ In Development ∗∗∗ Generate faults of okada sources from src mod mat files.

7.4 data_import/import_utils.py File Reference

Namespaces

- pyinsar.data_import.import_utils

Functions

- def pyinsar.data_import.import_utils.download_file (url, folder_path, username=None, password=None, filename=None)

  Download a file from a URL.

7.5 data_import/uavsar.py File Reference

Namespaces

- pyinsar.data_import.uavsar

Functions

- def pyinsar.data_import.uavsar.read_uavsar_metadata (in_file)

  Parse UAVSAR metadata.

7.6 output/export_georaster.py File Reference

Namespaces

- pyinsar.output.export_georaster

Functions

- def pyinsar.output.export_georaster.create_georaster_from_array (georaster_array, geotransform, projection, file_type='MEM', file_path='', data_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0., options=[ ])

  Create a GDAL georaster from a Numpy array.

7.7 output/plot_raster.py File Reference

Namespaces

- pyinsar.output.plot_raster

Functions

- def pyinsar.output.plot_raster.average_minmax_slices (array, axis=0)
- def pyinsar.output.plot_raster.plot_interactive_slicing (array, slice_index, model_array=None, axis=0, cmap='viridis', extent=None, clabel='', xlabel='', ylabel='', figsize=None, update_colorbar=False)
- def pyinsar.output.plot_raster.plot_interactive_multiple_slicing (array, axes, slice_indexes, model_array=None, cmap='viridis', update_colorbar=False, vmin=0., vmax=1., extent=None, clabel='', xlabel='', ylabel='', figsize=None)

7.8 processing/corrections/topography.py File Reference

Namespaces

- pyinsar.processing.corrections.topography

Functions

- def pyinsar.processing.corrections.topography.ellipsoidal_earth_slant_ranges (azimuth_time, latlon, orbit_interp, start_x, end_x, start_y, end_y)

  Compute slant ranges assuming no topography.

7.9 processing/corrections/troposphere.py File Reference

Namespaces

- pyinsar.processing.corrections.troposphere

Functions

- def pyinsar.processing.corrections.troposphere.vapor_pressure (T)

  Under development.
- def pyinsar.processing.corrections.troposphere.N (P, T, RH, k1=77.6, k2=23.3, k3=3.75E5)

  Under development.
- def pyinsar.processing.corrections.troposphere.N_h (h, P, T, RH, k1=77.6, k2=23.3, k3=3.75E5)

  Under development.
- def pyinsar.processing.corrections.troposphere.compute_delays (h, P, T, RH)

  Under development.

7.10 processing/data_fetcher/gdal.py File Reference

Classes

- class pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher

  Data fetcher for loading Images produced compatiable with GDAL.

Namespaces

- pyinsar.processing.data_fetcher.gdal

7.11 processing/data_fetcher/hdf_retriever.py File Reference

Classes

- class pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever

  Data fetcher for retrieving hdf image data made for training in convolutional neural networks.

Namespaces

- pyinsar.processing.data_fetcher.hdf_retriever

7.12 processing/deformation/elastic_halfspace/fault.py File Reference

Classes

- class pyinsar.processing.deformation.elastic_halfspace.fault.Fault

    ∗∗∗ *In Development* ∗∗∗ *Model a fault as a collection of small okada faults*

Namespaces

- pyinsar.processing.deformation.elastic_halfspace.fault

7.13 processing/deformation/elastic_halfspace/mogi.py File Reference

Namespaces

- pyinsar.processing.deformation.elastic_halfspace.mogi

Functions

- def pyinsar.processing.deformation.elastic_halfspace.mogi.compute_mogi_source_displacement (source_x, source_y, source_depth, source_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

7.14 processing/deformation/elastic_halfspace/okada.py File Reference

Namespaces

- pyinsar.processing.deformation.elastic_halfspace.okada

Functions

- def pyinsar.processing.deformation.elastic_halfspace.okada.I1 (xi, eta, q, delta, nu, R, X, d_tild)

  Okada's surface displacement.
- def pyinsar.processing.deformation.elastic_halfspace.okada.I2 (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I3 (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I4 (xi, eta, q, delta, nu, R, d_tild)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I5 (xi, eta, q, delta, nu, R, X, d_tild)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_strike (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_dip (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_x_tensile (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_strike (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_dip (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_y_tensile (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_strike (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_dip (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.f_z_tensile (xi, eta, q, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.chinnerys_notation (f, x, p, q, L, W, delta, nu)
- def pyinsar.processing.deformation.elastic_halfspace.okada.compute_okada_displacement (fault_centroid_x, fault_centroid_y, fault_centroid_depth, fault_strike, fault_dip, fault_length, fault_width, fault_rake, fault_slip, fault_open, poisson_ratio, xx_array, yy_array)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I1_int (xi, eta, z, y, delta, c, d, q, R)

  Okada's internal displacement.
- def pyinsar.processing.deformation.elastic_halfspace.okada.I2_int (xi, eta, z, y, delta, c, d, q, R)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I3_int (xi, eta, z, y, delta, c, d, q, R)
- def pyinsar.processing.deformation.elastic_halfspace.okada.I4_int (xi, eta, z, y, delta, c, d, q, R)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_strike (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_dip (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1_tensile (xi, eta, z, y, delta, c, alpha)

- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3_tensile (xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fA_3 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fB_3 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_1 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_2 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.fC_3 (displacement_type, xi, eta, z, y, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.chinnerys_notation_int (f, displacement_type, x, y, z, L, W, delta, c, alpha)
- def pyinsar.processing.deformation.elastic_halfspace.okada.compute_fault_internal_displacement_type (displacement_type, c, L, W, delta, U, alpha, xxx_array, yyy_array, zzz_array)
- def pyinsar.processing.deformation.elastic_halfspace.okada.compute_okada_internal_displacement (fault_centroid_x, fault_centroid_y, fault_centroid_depth, fault_strike, fault_dip, fault_length, fault_width, fault_rake, fault_slip, fault_open, poisson_ratio, xxx_array, yyy_array, depth_array)

7.15 processing/data_fetcher/okada.py File Reference

Classes

- class pyinsar.processing.data_fetcher.okada.DataFetcher

  Generates data from an Okada model.

Namespaces

- pyinsar.processing.data_fetcher.okada

7.16 processing/deformation/elastic_halfspace/pipe.py File Reference

Namespaces

- pyinsar.processing.deformation.elastic_halfspace.pipe

Functions

- def pyinsar.processing.deformation.elastic_halfspace.pipe.compute_closed_pipe_displacement (closed_pipe_x, closed_pipe_y, closed_pipe_depth_1, closed_pipe_depth_2, closed_pipe_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for a closed pipe.
- def pyinsar.processing.deformation.elastic_halfspace.pipe.compute_open_pipe_displacement (open_pipe_x, open_pipe_y, open_pipe_depth_0, open_pipe_depth_1, open_pipe_depth_2, open_pipe_radius, poisson_ratio, pressurization, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for an open pipe.

7.17 processing/deformation/elastic_halfspace/surface_load.py File Reference

Namespaces

- pyinsar.processing.deformation.elastic_halfspace.surface_load

Functions

- def pyinsar.processing.deformation.elastic_halfspace.surface_load.compute_uniform_disk_load_displacement (disk_x, disk_y, disk_radius, poisson_ratio, pressure, shear_modulus, xx_array, yy_array)

  Compute the surface displacements for a uniform disk load.

7.18 processing/discovery/classify_cnn.py File Reference

Classes

- class pyinsar.processing.discovery.ClassifyCNN

  Train a CNN.

Namespaces

- pyinsar.processing.discovery.classify_cnn

7.19 processing/discovery/coherence.py File Reference

Classes

- class pyinsar.processing.discovery.Coherence

  Calculate coherence between single-look complex SAR images.

Namespaces

- pyinsar.processing.discovery.coherence

7.20 processing/discovery/coregister.py File Reference

Classes

- class pyinsar.processing.discovery.Coregister

  ∗∗∗ In Devolopment ∗∗∗ Pipeline item to coregister images

Namespaces

- pyinsar.processing.discovery.coregister

7.21 processing/discovery/deburst.py File Reference

Classes

- class pyinsar.processing.discovery.deburst.Deburst

  Debursts Sentinel-1 TOPSAR data.

Namespaces

- pyinsar.processing.discovery.deburst

7.22 processing/discovery/deformation_to_phase.py File Reference

Classes

- class pyinsar.processing.discovery.DeformationToPhase

  Convert deformation to phas.

Namespaces

- pyinsar.processing.discovery.deformation_to_phase

7.23 processing/discovery/flat_earth.py File Reference

Classes

- class pyinsar.processing.discovery.FlatEarth

  ∗∗∗ *In Development* ∗∗∗ *Remove flat Earth contribution from interferogram*

Namespaces

- pyinsar.processing.discovery.flat_earth

7.24 processing/discovery/interferogram.py File Reference

Classes

- class pyinsar.processing.discovery.interferogram.Interferogram

  Create Inteferogram from SLC data.

Namespaces

- pyinsar.processing.discovery.interferogram

7.25 processing/discovery/los_deformation.py File Reference

Classes

- class pyinsar.processing.discovery.LOS_Deformation

  *** *In Development* ***

Namespaces

- pyinsar.processing.discovery.los_deformation

7.26 processing/discovery/rotate_squares.py File Reference

Classes

- class pyinsar.processing.discovery.RotateSquares

  Generate new images by rotating subsections of data defined by Shapely squares.

Namespaces

- pyinsar.processing.discovery.rotate_squares

Functions

- def pyinsar.processing.discovery.rotateSquare (image, square, angle, order)

  Rotate a subsection of an image defined by a shapely square.

7.27 processing/discovery/shown_cnn_classes.py File Reference

Classes

- class pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses

  Dispay CNN Classifications on segments of an image.

Namespaces

- pyinsar.processing.discovery.shown_cnn_classes

7.28 processing/discovery/temporal_decorrelation.py File Reference

Classes

- class pyinsar.processing.discovery.TemporalDecorrelation

  Pipeline item to add temporal decorrelation to some phase.

Namespaces

- pyinsar.processing.discovery.temporal_decorrelation

7.29 processing/discovery/train_cnn.py File Reference

Classes

- class pyinsar.processing.discovery.TrainCNN

  Train a CNN.

Namespaces

- pyinsar.processing.discovery.train_cnn

7.30 processing/discovery/wrap_phase.py File Reference

Classes

- class pyinsar.processing.discovery.WrapPhase

  Pipeline Item that wraps phase.

Namespaces

- pyinsar.processing.discovery.wrap_phase

7.31 processing/geography/coordinates.py File Reference

Namespaces

- pyinsar.processing.geography.coordinates

Functions

- def pyinsar.processing.geography.coordinates.transform_to_pixel_coordinates (x, y, x_min, x_max, y_min, y_max, array_width, array_height)

  Array coordinates.
- def pyinsar.processing.geography.coordinates.transform_to_geographic_coordinates (u, v, x_min, x_max, y_min, y_max, array_width, array_height)

  Transform some pixel coordinates in an array to geographic coordinates.
- def pyinsar.processing.geography.coordinates.compute_x_and_y_coordinates_maps (x_min, x_max, y_min, y_max, array_width, array_height)

  Compute an array of x and y coordinates based on an extent and array shape.
- def pyinsar.processing.geography.coordinates.extract_subgeoarray (georaster_array, georaster_extent, x_min, x_max, y_min, y_max, center_extent=False)
- def pyinsar.processing.geography.coordinates.sample_nd_array (array, subarray_shape, steps=(1, 1))
- def pyinsar.processing.geography.coordinates.sample_2d_array (array, subarray_shape, steps=(1, 1), is_shape_centered=False)
- def pyinsar.processing.geography.coordinates.sample_2d_multiarray (array, subarray_shape, steps=(1, 1))
- def pyinsar.processing.geography.coordinates.reproject_point (lon, lat, old_projection_EPSG=None, old_projection_wkt=None, old_projection_utm=None, new_projection_EPSG=None, new_projection_wkt=None, new_projection_utm=None)

  Projection.
- def pyinsar.processing.geography.coordinates.find_utm_area (longitude, latitude)

  Find the UTM code and hemisphere from the longitude and latitude of a point.
- def pyinsar.processing.geography.coordinates.reproject_georaster (georaster, new_cell_sizes, new_projection_EPSG=None, new_projection_wkt=None, new_projection_utm=None, new_extent=None, interpolation_method=gdal.GRA_Cubic, file_type='MEM', file_path='', data_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0., options=[ ])

  Change the projection of a GDAL georaster.
- def pyinsar.processing.geography.coordinates.georaster_vertical_datum_shift (georaster, old_datum_proj4='+proj=longlat+datum=WGS84+no_defs+geoidgrids=egm96_15.gtx', new_datum_proj4='+proj=longlat+datum=WGS84+no_defs', file_type='MEM', file_path='', data_type=gdal.GDT_Float64, no_data_value=-99999., scale=1., offset=0.)

Variables

- pyinsar.processing.geography.coordinates.nopython

  Extract all the possible sub-arrays that do not contain any NaN.
- pyinsar.processing.geography.coordinates.True
- pyinsar.processing.geography.coordinates.nogil
- pyinsar.processing.geography.coordinates.parallel

7.32 processing/geography/geodesy.py File Reference

Namespaces

- pyinsar.processing.geography.geodesy

Functions

- def pyinsar.processing.geography.geodesy.compute_great_circle_distance_and_bearing (rad_longitude_1, rad_latitude_1, rad_longitude_2, rad_latitude_2, planet_radius)

  Geodesy on a sphere.
- def pyinsar.processing.geography.geodesy.compute_lonlat_from_distance_bearing (rad_longitude_1, rad_latitude_1, distance, rad_bearing, planet_radius)
- def pyinsar.processing.geography.geodesy.direct_vincenty_formula (rad_lon_1, rad_lat_1, distance, rad_bearing_1, a, f, eps=1e-12)
- def pyinsar.processing.geography.geodesy.direct_vincenty_formula_for_array (rad_longitude_1_array, rad_latitude_1_array, distance_array, rad_bearing_1, a, f, eps=1e-12)
- def pyinsar.processing.geography.geodesy.update_lambda (Lambda, reduced_rad_lat_1, reduced_rad_lat_2, diff_lon, f)
- def pyinsar.processing.geography.geodesy.inverse_vincenty_formula (rad_lon_1, rad_lat_1, rad_lon_2, rad_lat_2, a, f, eps=1e-12, max_iter=200)
- def pyinsar.processing.geography.geodesy.inverse_vincenty_formula_for_array (rad_longitude_1, rad_latitude_1, rad_longitude_2_array, rad_latitude_2_array, a, f, eps=1e-12, max_iter=200)
- def pyinsar.processing.geography.geodesy.compute_point_to_line_distance_on_ellipsoid (rad_point_lon, rad_point_lat, rad_geodesic_origin_lon, rad_geodesic_origin_lat, rad_geodesic_bearing, a, f, eps=1e-12, max_iter=200)
- def pyinsar.processing.geography.geodesy.compute_point_to_line_distance_for_array (rad_longitude_1, rad_latitude_1, rad_longitude_2_array, rad_latitude_2_array, rad_bearing, a, f, eps=1e-12, max_iter=200)

Variables

- pyinsar.processing.geography.geodesy.nopython

  Geodesy on an oblate spheroid.

7.33 processing/geography/geomorphometry.py File Reference

Namespaces

- pyinsar.processing.geography.geomorphometry

Functions

- def pyinsar.processing.geography.geomorphometry.add_symmetric_border (array, border_size=1)
- def pyinsar.processing.geography.geomorphometry.compute_gradient_at_cell (array, j, i, grid_yx_spacing, axis=1)
- def pyinsar.processing.geography.geomorphometry.compute_horne_slope (array, grid_yx_spacing)

Variables

- pyinsar.processing.geography.geomorphometry.nopython

  Add a symmetric border to a 2D array.

7.34 processing/instruments/sentinel.py File Reference

Classes

- class pyinsar.processing.instruments.sentinel.RampPolynomial

  Polynomial used for quantities relating to deramping sentinel.
- class pyinsar.processing.instruments.sentinel.SentinelRamp

  Calcuate the combined ramp and modulated phase in Sentinel.

Namespaces

- pyinsar.processing.instruments.sentinel

Functions

- def pyinsar.processing.instruments.sentinel.transform_slc (slc, deramped_phase, transformation_matrix)
- def pyinsar.processing.instruments.sentinel.find_overlapping_valid_lines (metadata_tree)

  Determine which lines between bursts overlap.
- def pyinsar.processing.instruments.sentinel.get_valid_lines (metadata_tree, per_burst=False)

  Retrieve all lines that contain some valid data.
- def pyinsar.processing.instruments.sentinel.select_valid_lines (data, tree, cut=True)

  Extract burst information from SLC.
- def pyinsar.processing.instruments.sentinel.retrieve_azimuth_time (in_tree)

  Retrieves the zero azimuth time for all the lines in the data.
- def pyinsar.processing.instruments.sentinel.read_geolocation (tree)

  Read in geolocation data.
- def pyinsar.processing.instruments.sentinel.update_geolocation_lines (tree, azimuth_times, geolocation_data)

  Update which line is associated with geolocation data using azimuth times.
- def pyinsar.processing.instruments.sentinel.get_sentinel_extents (geolocation, offset=0.0)

  Get the extents (latitude and longitude) of a sentinel-1 image given its geolocation information.

7.35 data_import/sentinel.py File Reference

Namespaces

- pyinsar.data_import.sentinel

Functions

- def pyinsar.data_import.sentinel.parse_satellite_data (in_satellite_file)

  Parse Sentinel satellite data.
- def pyinsar.data_import.sentinel.get_url_precise_orbit (product_name)
- def pyinsar.data_import.sentinel.download_precise_orbits (product_folder, orbit_folder, username, password)

  Download the precise orbits for all the Sentinel-1 products in a folder.
- def pyinsar.data_import.sentinel.download_products (product_names, product_folder, base_url='https://datapool.asf.alaska.edu/SLC', use_vertex=True, username=None, password=None)

  Download Sentinel-1 products in a folder.

7.36 processing/isce/input_file.py File Reference

Namespaces

- pyinsar.processing.isce.input_file

Functions

- def pyinsar.processing.isce.input_file.create_product_xml (xml_path, product_path, product_type='master', product_output_path=None, product_orbit_path=None, product_auxiliary_data_path=None, do_add=True)

  Create the xml file defining a Sentinel-1 product for processing with ISCE.
- def pyinsar.processing.isce.input_file.create_topsApp_xml (xml_folder_path, master_path, slave_path, master_output_path=None, slave_output_path=None, master_orbit_path=None, slave_orbit_path=None, master_auxiliary_data_path=None, slave_auxiliary_data_path=None, do_unwrap=True, unwrapper_name='snaphu_mcf', xml_filename='topsApp.xml')

  Create the topsApp.xml file for processing Sentinel-1 data with ISCE.
- def pyinsar.processing.isce.input_file.prepare_topsApps (product_paths, result_folder_path, orbit_path=None, auxiliary_data_path=None, do_unwrap=True, unwrapper_name='snaphu_mcf')

7.37 processing/machine_learning/geostatistics/direct_sampling.py File Reference

Namespaces

- pyinsar.processing.machine_learning.geostatistics.direct_sampling

Functions

- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_neighborhood_lag_vectors (neighborhood_shape, grid_yx_spacing, delta)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_neighborhoods (simulation_array, data_weight_array, cell_j, cell_i, lag_vectors, lag_distances, max_number_data, max_density_data, neighborhood_shape, rotation_angle_rad, scaling_factor, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_continuous_distance (training_image_array, ti_j, ti_i, ti_ranges_max, neighbor_indexes, neighbor_values, neighbor_numbers, min_distances, var_k, max_non_matching_proportion, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.compute_discrete_distance (training_image_array, ti_j, ti_i, neighbor_indexes, neighbor_values, neighbor_numbers, min_distances, var_k, max_non_matching_proportion, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.find_closest_cell_in_training_image (training_image_array, ti_ranges_max, ti_indices, ti_index, neighbor_indexes, neighbor_values, neighbor_numbers, distance_thresholds, max_non_matching_proportion, ti_fraction, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.prepare_training_image (array, variable_types)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.is_any_equal (list_1, value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.is_any_nan (list_1)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.run_ds (data_array, training_image_array, variable_types, distance_thresholds, ti_fraction, max_number_data, max_density_data, neighborhood_shape=(math.inf, math.inf), grid_yx_spacing=(1., 1.), delta=0., conditioning_data_weight=1., max_non_matching_proportion=1., start_parameter_reduction=1, reduction_factor=1, rotation_angle_array=np.empty((1, 1)), scaling_factor_array=np.empty((1, 1, 1)), number_postproc=0, postproc_factor=1, number_realizations=1, path_type=PathType.RANDOM, seed=100, no_data_value=-99999)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.simulate_ds_realization (data_array, data_weight_array, training_image_array, ti_ranges_max, ti_indices, distance_thresholds, ti_fraction, max_number_data, max_density_data, lag_vectors, lag_distances, neighborhood_shape, max_non_matching_proportion, start_parameter_reduction, reduction_factor, rotation_angle_array, scaling_factor_array, number_postproc, postproc_factor, path_type, seed, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.direct_sampling.run_parallel_ds (data_array, training_image_array, variable_types, distance_thresholds, ti_fraction, max_number_data, max_density_data, neighborhood_shape=(math.inf, math.inf), grid_yx_spacing=(1., 1.), delta=0., conditioning_data_weight=1., max_non_matching_proportion=1., start_parameter_reduction=1, reduction_factor=1, rotation_angle_array=np.empty((1, 1)), scaling_factor_array=np.empty((1, 1, 1)), number_postproc=0, postproc_factor=1, number_realizations=1, path_type=PathType.RANDOM, seed=100, no_data_value=-99999)

Variables

- pyinsar.processing.machine_learning.geostatistics.direct_sampling.nopython

  Compute the lag vectors for the neighborhood, assuming a regular grid.

7.38 processing/machine_learning/geostatistics/geostatistics_utils.py File Reference

Classes

- class pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType
- class pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType

Namespaces

- pyinsar.processing.machine_learning.geostatistics.geostatistics_utils

Functions

- def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.unflatten_index (flattened_index, array_shape)
- def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.standardize (x)

  Reduce and center a float or array.
- def pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.normalize (x)

  Reduce and center a float or array.

Variables

- pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.nopython

  Unflatten an index for a 2D array.
- pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.True
- pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.nogil

7.39 processing/machine_learning/geostatistics/sequential_gaussian_simulation.py File Reference

Classes

- class pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod

Namespaces

- pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation

Functions

- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.merge_secondary_data (secondary_data_array, correlations_with_primary, correlations_between_secondary)

  Merging secondary data.
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_euclidean_distance (cell_1, cell_2)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_axis_aligned_ellipse_range (neighborhood_range, neighborhood_azimuth_rad)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_axis_aligned_neighborhood_shape (neighborhood_range, neighborhood_azimuth, grid_yx_spacing)

- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_neighborhood_template (neighborhood_range, grid_yx_spacing, vario_models, vario_sills, vario_ranges, vario_azimuth_rad, rotation_matrix, eps=0.0001)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_neighborhood (cell_index, simulation_array, neighborhood_template, max_number_data, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_values_matrix (neighborhood, simulation_array)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_data_to_data_matrix (kriging_method, cell_index, neighborhood, correlation_template, secondary_data_weight)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.get_data_to_unknown_matrix (kriging_method, cell_index, neighborhood, correlation_template, secondary_data_weight)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.solve_kriging_system (cell_index, neighborhood, simulation_array, primary_mean, primary_variance, correlation_template, secondary_data_weight, secondary_data_mean, secondary_data_array)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.run_sgs (data_array, grid_yx_spacing, vario_models, vario_sills, vario_azimuth, vario_ranges, number_realizations=1, path_type=PathType.RANDOM, kriging_method=KrigingMethod.SIMPLE, neighborhood_range=(math.nan, math.nan), max_number_data=12, secondary_data_weight=math.nan, secondary_data_array=np.empty((1, 1)), seed=100, no_data_value=-99999.)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.simulate_sgs_realization (data_array, path_type, primary_mean, primary_variance, neighborhood_template, correlation_template, max_number_data, secondary_data_weight, secondary_data_array, seed, no_data_value)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.run_parallel_sgs (data_array, grid_yx_spacing, vario_models, vario_sills, vario_azimuth, vario_ranges, number_realizations=1, path_type=PathType.RANDOM, kriging_method=KrigingMethod.SIMPLE, neighborhood_range=(math.nan, math.nan), max_number_data=12, secondary_data_weight=math.nan, secondary_data_array=np.empty((1, 1)), seed=100, nb_threads=4, no_data_value=-99999.)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.inverse_standard_normal_cdf (x)

  Data transform.
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.compute_averaged_cumulative_distribution_from_array (value_array)
- def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.normal_score_tranform (value_array)

  Transform the values of an array to a normal distribution.

Variables

- pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.nopython

  Sequential Gaussian Simulation (SGS)
- pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.True
- pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.nogil

7.40 processing/machine_learning/geostatistics/variogram.py File Reference

Classes

- class pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel

  2D theoretical variogram

Namespaces

- pyinsar.processing.machine_learning.geostatistics.variogram

Functions

- def pyinsar.processing.machine_learning.geostatistics.variogram.compute_experimental_variogram (value_↩ array, grid_yx_spacing, number_of_lags, lag_unit_distance, tolerance=None, sampling=1., no_data_value=-99999)
- def pyinsar.processing.machine_learning.geostatistics.variogram.nugget_variogram (reduced_distance, variance↩ _contribution)
- def pyinsar.processing.machine_learning.geostatistics.variogram.gaussian_variogram (reduced_distance, variance_contribution)
- def pyinsar.processing.machine_learning.geostatistics.variogram.spherical_variogram (reduced_distance, variance_contribution)
- def pyinsar.processing.machine_learning.geostatistics.variogram.exponential_variogram (reduced_distance, variance_contribution)
- def pyinsar.processing.machine_learning.geostatistics.variogram.compute_variogram (delta_y, delta_x, vario_↩ models, vario_sills, vario_ranges, rotation_matrix)
- def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_gaussian_variogram (distance, vario_range, variance_contribution)

    Vectorized theoretical variogram.
- def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_spherical_variogram (distance, vario_range, variance_contribution)

    Compute the value of a variogram with a spherical model.
- def pyinsar.processing.machine_learning.geostatistics.variogram.vectorized_exponential_variogram (distance, vario_range, variance_contribution)

    Compute the value of a variogram with an exponential model.
- def pyinsar.processing.machine_learning.geostatistics.variogram.map_2D_variogram (vario_models, vario_sills, vario_azimuth, vario_ranges, neighborhood_range, map_shape, grid_spacing)
- def pyinsar.processing.machine_learning.geostatistics.variogram.compute_range_variogram (deltas_y, deltas_x, vario_models, vario_sills, vario_ranges, vario_azimuth=0.)

Variables

- pyinsar.processing.machine_learning.geostatistics.variogram.nopython

    2D experimental variogram
- pyinsar.processing.machine_learning.geostatistics.variogram.True
- pyinsar.processing.machine_learning.geostatistics.variogram.nogil

7.41 processing/utilities/ann.py File Reference

Namespaces

- pyinsar.processing.utilities.ann

Functions

- def pyinsar.processing.utilities.ann.buildCNN (image_height, image_width, model_dir, rate=0.01, config=None)

  Build a convolutional neural network.
- def pyinsar.processing.utilities.ann.train (image_data, image_labels, model_dir, batch_size, num_epochs, max←↩_batches=None, status_line_rate=50, target='', shuffle=True, config=None)

  Train neural network.
- def pyinsar.processing.utilities.ann.classify (image_data, model_dir, batch_size=2000, config=None)

  Classify data.
- def pyinsar.processing.utilities.ann.length_after_valid_window (length, window, stride)

  Length of dimension after convolving using the padding type 'valid' or using max pooling.
- def pyinsar.processing.utilities.ann.shuffleTrainingData (data, labels)

  Shuffles data.
- def pyinsar.processing.utilities.ann.restoreGraph (model_dir)

  Restore a network.

7.42 processing/utilities/deformations.py File Reference

Namespaces

- pyinsar.processing.utilities.deformations

Functions

- def pyinsar.processing.utilities.deformations.calc_bounding_box (image)

  Calculate bounding box of an object in an image.
- def pyinsar.processing.utilities.deformations.determine_deformation_bounding_box (deformations)

  Determine bounds around a deformation.
- def pyinsar.processing.utilities.deformations.determine_x_y_bounds (deformations, x_array, y_array, offset=5000)

  Determine the x and y positions that bound a deformation.

7.43 processing/utilities/generic.py File Reference

Classes

- class pyinsar.processing.utilities.generic.OrbitInterpolation

  Class for interpolating satellite positions.
- class pyinsar.processing.utilities.generic.FindNearestPixel

  Find the nearest given a time.

Namespaces

- pyinsar.processing.utilities.generic

Functions

- def pyinsar.processing.utilities.generic.get_image_extents (geotransform, shape)

  Get extents of in projection coordinates.
- def pyinsar.processing.utilities.generic.proj4StringToDictionary (proj4_string)

  Convert a proj4 string into a dictionary.
- def pyinsar.processing.utilities.generic.sorted_alphanumeric (l)

  Sort a list of strings with numbers.
- def pyinsar.processing.utilities.generic.phase_shift (data, phase)

  Apply a phase shift to data.
- def pyinsar.processing.utilities.generic.find_closest_time (time, date)

  Find the closest time to a date.
- def pyinsar.processing.utilities.generic.rotate (col_vectors, az, ay, ax, dtype=np.float64)

  Rotate 3 dimensional column vectors.
- def pyinsar.processing.utilities.generic.translate (col_vectors, delta_x, delta_y, delta_z)

  Translate 3 dimensional column vectors.
- def pyinsar.processing.utilities.generic.coherence (s1, s2, window, topo_phase=0)

  This function computes the coherence between two SLCs.
- def pyinsar.processing.utilities.generic.scale_image (input_data, vmin=None, vmax=None)
- def pyinsar.processing.utilities.generic.keypoints_align (img1, img2, max_matches=40, invert=True)

  ∗∗∗ *In Development* ∗∗∗ *Determine transformation matrix for aligning images*
- def pyinsar.processing.utilities.generic.subarray_slice (index, num_items)

  Returns a slice that selects for selecting a chunk out of an array.
- def pyinsar.processing.utilities.generic.find_data_asf (lat, lon, processingLevel='SLC', platform='Sentinel-1A, Sentinel, B, kwargs)

  Search Alaska Satellite Facility for data.
- def pyinsar.processing.utilities.generic.select_max_matched_data (sentinel_data_list)

  Select the data that can be combined into an interferogram.
- def pyinsar.processing.utilities.generic.match_data (sentinel_data_list)

  Seperate into sets of overlapping data.
- def pyinsar.processing.utilities.generic.find_earthquake_pairs (organized_data, date)

  Select image pairs around a specified date.
- def pyinsar.processing.utilities.generic.generateMatplotlibRectangle (extent, kwargs)

  Generate a matplotlib rectangle from a extents.
- def pyinsar.processing.utilities.generic.project_insar_data (in_dataset, lon_center, lat_center, interpolation=gdal.↩ GRA_Cubic, no_data_value=np.nan, data_type=gdal.GDT_Float64)

  Project InSAR data using GDAL.

7.44 processing/utilities/insar_simulator_utils.py File Reference

Namespaces

- pyinsar.processing.utilities.insar_simulator_utils

Functions

- def pyinsar.processing.utilities.insar_simulator_utils.wrap (x, to_2pi=False)

  Wrap a float or an array.
- def pyinsar.processing.utilities.insar_simulator_utils.crop_array_from_center (array, crop_shape)

  Crop an array along its borders.
- def pyinsar.processing.utilities.insar_simulator_utils.mask_deformation (deformation, threshold_function=threshold↩ _li)

  Mask image using a threshold function.
- def pyinsar.processing.utilities.insar_simulator_utils.calc_bounding_box (image, threshold_function=threshold_li)

  Calcluate the bounding box around an image using the li threshold.
- def pyinsar.processing.utilities.insar_simulator_utils.retrieve_bounds (thresh_image)

  Retrieve the bounds of an image that has been thesholded.
- def pyinsar.processing.utilities.insar_simulator_utils.crop_nans (image)

  Shrink image by removing nans.
- def pyinsar.processing.utilities.insar_simulator_utils.determine_deformation_bounding_box (deformations, largest_box=True, kwargs)

  Calculate the extent of the deformation in image coordinates.
- def pyinsar.processing.utilities.insar_simulator_utils.determine_x_y_bounds (deformations, x_array, y_array, offset=5000, kwargs)

  Determine the x and y coordinates of the extent of the deformation.
- def pyinsar.processing.utilities.insar_simulator_utils.generate_interferogram_from_deformation (track_↩ angle, min_ground_range_1, height_1, is_right_looking, wavelength, k, deformation, xx, yy, projected_↩ topography=None, min_ground_range_2=None, height_2=None)

  Generate an interferogram from deformations.
- def pyinsar.processing.utilities.insar_simulator_utils.old_generate_interferogram_from_deformation (track_angle, min_ground_range, height, is_right_looking, wavelength, k, deformation, xx, yy, projected_topography=None)

  Generate an interferogram from deformations.
- def pyinsar.processing.utilities.insar_simulator_utils.change_in_range_to_phase (los_deformation, wavelength, k=2)

  Compute phase from change in range.
- def pyinsar.processing.utilities.insar_simulator_utils.phase_to_change_in_range (phase, wavelength, k=2)

  Compute change in range from phase.

7.45 processing/utilities/machine_learning.py File Reference

Classes

- class pyinsar.processing.utilities.machine_learning.DataRetriever

  Class for retrieving data from an hdf file.

Namespaces

- pyinsar.processing.utilities.machine_learning

Functions

- def pyinsar.processing.utilities.machine_learning.divide_into_squares (image, size, stride)

  Create many patches from an image.
- def pyinsar.processing.utilities.machine_learning.generate_minimum_ground_range_limits (satellite_height, incidence_ranges, image_size)

  Determine the limits of minimum ground ranges of a satellite pass.
- def pyinsar.processing.utilities.machine_learning.generate_phase_samples_from_looks_and_ranges (deformation↩_list, xx, yy, satellite_height, track_angles, minimum_ground_ranges, size=(100, 100), dtype=np.float32)

  Generates different possible phases from a list of deformations due to different track angles and groud ranges.
- def pyinsar.processing.utilities.machine_learning.generate_phase_samples (deformation, satellite_height, radar_wavelength, cell_size, image_size, stride=20)

  ***In Development** Generate phase samples by tiling an array of deformations*
- def pyinsar.processing.utilities.machine_learning.rotate_image_list (in_image_extents, in_image_list, progress=True)

  Rotate input images 0, 90, 180, and 270 degrees.

Index