

PyInSAR

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	pyinsar Namespace Reference	9
5.2	pyinsar.data_import Namespace Reference	9
5.3	pyinsar.data_import.import_georaster Namespace Reference	9
5.3.1	Function Documentation	10
5.3.1.1	get_georaster_array()	10
5.3.1.2	get_georaster_extent()	10
5.3.1.3	open_georaster()	11
5.3.1.4	print_georaster_info()	11
5.4	pyinsar.data_import.import_raster Namespace Reference	11
5.4.1	Function Documentation	12

5.4.1.1	<code>open_gacos_tropospheric_delays()</code>	12
5.4.1.2	<code>open_sgems_file()</code>	12
5.4.1.3	<code>open_sgems_file_from_url()</code>	12
5.4.1.4	<code>read_rsc_header_file()</code>	13
5.5	<code>pyinsar.data_import.import_srcmod</code> Namespace Reference	13
5.5.1	Function Documentation	13
5.5.1.1	<code>read_srcmod_data()</code>	13
5.6	<code>pyinsar.data_import.import_utils</code> Namespace Reference	14
5.6.1	Function Documentation	14
5.6.1.1	<code>download_file()</code>	14
5.7	<code>pyinsar.data_import.sentinel</code> Namespace Reference	15
5.7.1	Function Documentation	15
5.7.1.1	<code>download_precise_orbits()</code>	15
5.7.1.2	<code>download_products()</code>	15
5.7.1.3	<code>get_url_precise_orbit()</code>	16
5.7.1.4	<code>parse_satellite_data()</code>	16
5.8	<code>pyinsar.data_import.uavsar</code> Namespace Reference	17
5.8.1	Function Documentation	17
5.8.1.1	<code>read_uavsar_metadata()</code>	17
5.9	<code>pyinsar.output</code> Namespace Reference	17
5.10	<code>pyinsar.output.export_georaster</code> Namespace Reference	17
5.10.1	Function Documentation	18
5.10.1.1	<code>create_georaster_from_array()</code>	18
5.11	<code>pyinsar.output.plot_raster</code> Namespace Reference	18
5.11.1	Function Documentation	19
5.11.1.1	<code>average_minmax_slices()</code>	19
5.11.1.2	<code>plot_interactive_multiple_slicing()</code>	19
5.11.1.3	<code>plot_interactive_slicing()</code>	19

5.12 pyinsar.processing Namespace Reference	20
5.13 pyinsar.processing.corrections Namespace Reference	20
5.14 pyinsar.processing.corrections.topography Namespace Reference	20
5.14.1 Function Documentation	20
5.14.1.1 ellipsoidal_earth_slant_ranges()	20
5.15 pyinsar.processing.corrections.troposphere Namespace Reference	21
5.15.1 Function Documentation	21
5.15.1.1 compute_delays()	21
5.15.1.2 N()	22
5.15.1.3 N_h()	22
5.15.1.4 vapor_pressure()	22
5.16 pyinsar.processing.data_fetcher Namespace Reference	22
5.17 pyinsar.processing.data_fetcher.gdal Namespace Reference	23
5.18 pyinsar.processing.data_fetcher.hdf_retriever Namespace Reference	23
5.19 pyinsar.processing.data_fetcher.okada Namespace Reference	23
5.20 pyinsar.processing.deformation Namespace Reference	23
5.21 pyinsar.processing.deformation.elastic_halfspace Namespace Reference	23
5.22 pyinsar.processing.deformation.elastic_halfspace.fault Namespace Reference	24
5.23 pyinsar.processing.deformation.elastic_halfspace.mogi Namespace Reference	24
5.23.1 Function Documentation	24
5.23.1.1 compute_mogi_source_displacement()	24
5.24 pyinsar.processing.deformation.elastic_halfspace.okada Namespace Reference	24
5.24.1 Function Documentation	26
5.24.1.1 chinnerys_notation()	26
5.24.1.2 chinnerys_notation_int()	26
5.24.1.3 compute_fault_internal_displacement_type()	27
5.24.1.4 compute_okada_displacement()	27
5.24.1.5 compute_okada_internal_displacement()	27

5.24.1.6 <code>f_x_dip()</code>	28
5.24.1.7 <code>f_x_strike()</code>	28
5.24.1.8 <code>f_x_tensile()</code>	28
5.24.1.9 <code>f_y_dip()</code>	28
5.24.1.10 <code>f_y_strike()</code>	29
5.24.1.11 <code>f_y_tensile()</code>	29
5.24.1.12 <code>f_z_dip()</code>	29
5.24.1.13 <code>f_z_strike()</code>	29
5.24.1.14 <code>f_z_tensile()</code>	30
5.24.1.15 <code>fA_1()</code>	30
5.24.1.16 <code>fA_1_dip()</code>	30
5.24.1.17 <code>fA_1_strike()</code>	30
5.24.1.18 <code>fA_1_tensile()</code>	31
5.24.1.19 <code>fA_2()</code>	31
5.24.1.20 <code>fA_2_dip()</code>	31
5.24.1.21 <code>fA_2_strike()</code>	31
5.24.1.22 <code>fA_2_tensile()</code>	32
5.24.1.23 <code>fA_3()</code>	32
5.24.1.24 <code>fA_3_dip()</code>	32
5.24.1.25 <code>fA_3_strike()</code>	32
5.24.1.26 <code>fA_3_tensile()</code>	33
5.24.1.27 <code>fB_1()</code>	33
5.24.1.28 <code>fB_1_dip()</code>	33
5.24.1.29 <code>fB_1_strike()</code>	33
5.24.1.30 <code>fB_1_tensile()</code>	34
5.24.1.31 <code>fB_2()</code>	34
5.24.1.32 <code>fB_2_dip()</code>	34
5.24.1.33 <code>fB_2_strike()</code>	34

5.24.1.34 <code>fB_2_tensile()</code>	35
5.24.1.35 <code>fB_3()</code>	35
5.24.1.36 <code>fB_3_dip()</code>	35
5.24.1.37 <code>fB_3_strike()</code>	35
5.24.1.38 <code>fB_3_tensile()</code>	36
5.24.1.39 <code>fC_1()</code>	36
5.24.1.40 <code>fC_1_dip()</code>	36
5.24.1.41 <code>fC_1_strike()</code>	36
5.24.1.42 <code>fC_1_tensile()</code>	37
5.24.1.43 <code>fC_2()</code>	37
5.24.1.44 <code>fC_2_dip()</code>	37
5.24.1.45 <code>fC_2_strike()</code>	37
5.24.1.46 <code>fC_2_tensile()</code>	38
5.24.1.47 <code>fC_3()</code>	38
5.24.1.48 <code>fC_3_dip()</code>	38
5.24.1.49 <code>fC_3_strike()</code>	38
5.24.1.50 <code>fC_3_tensile()</code>	39
5.24.1.51 <code>I1()</code>	39
5.24.1.52 <code>I1_int()</code>	39
5.24.1.53 <code>I2()</code>	40
5.24.1.54 <code>I2_int()</code>	40
5.24.1.55 <code>I3()</code>	40
5.24.1.56 <code>I3_int()</code>	41
5.24.1.57 <code>I4()</code>	41
5.24.1.58 <code>I4_int()</code>	41
5.24.1.59 <code>I5()</code>	41
5.25 <code>pyinsar.processing.deformation.elastic_halfspace.pipe</code> Namespace Reference	42
5.25.1 Function Documentation	42

5.25.1.1	<code>compute_closed_pipe_displacement()</code>	42
5.25.1.2	<code>compute_open_pipe_displacement()</code>	43
5.26	<code>pyinsar.processing.deformation.elastic_halfspace.surface_load</code> Namespace Reference	43
5.26.1	Function Documentation	44
5.26.1.1	<code>compute_uniform_disk_load_displacement()</code>	44
5.27	<code>pyinsar.processing.discovery</code> Namespace Reference	44
5.28	<code>pyinsar.processing.discovery.classify_cnn</code> Namespace Reference	45
5.29	<code>pyinsar.processing.discovery.coherence</code> Namespace Reference	45
5.30	<code>pyinsar.processing.discovery.coregister</code> Namespace Reference	45
5.31	<code>pyinsar.processing.discovery.deburst</code> Namespace Reference	45
5.32	<code>pyinsar.processing.discovery.deformation_to_phase</code> Namespace Reference	46
5.33	<code>pyinsar.processing.discovery.flat_earth</code> Namespace Reference	46
5.34	<code>pyinsar.processing.discovery.interferogram</code> Namespace Reference	46
5.35	<code>pyinsar.processing.discovery.los_deformation</code> Namespace Reference	46
5.36	<code>pyinsar.processing.discovery.rotate_squares</code> Namespace Reference	46
5.36.1	Function Documentation	47
5.36.1.1	<code>rotateSquare()</code>	47
5.37	<code>pyinsar.processing.discovery.shown_cnn_classes</code> Namespace Reference	47
5.38	<code>pyinsar.processing.discovery.temporal_decorrelation</code> Namespace Reference	47
5.39	<code>pyinsar.processing.discovery.train_cnn</code> Namespace Reference	47
5.40	<code>pyinsar.processing.discovery.wrap_phase</code> Namespace Reference	48
5.41	<code>pyinsar.processing.geography</code> Namespace Reference	48
5.42	<code>pyinsar.processing.geography.coordinates</code> Namespace Reference	48
5.42.1	Function Documentation	49
5.42.1.1	<code>compute_x_and_y_coordinates_maps()</code>	49
5.42.1.2	<code>extract_subgeoarray()</code>	49
5.42.1.3	<code>find_utm_area()</code>	50
5.42.1.4	<code>georaster_vertical_datum_shift()</code>	50

5.42.1.5	reproject_georaster()	51
5.42.1.6	reproject_point()	51
5.42.1.7	sample_2d_array()	52
5.42.1.8	sample_2d_multiarray()	52
5.42.1.9	sample_nd_array()	52
5.42.1.10	transform_to_geographic_coordinates()	53
5.42.1.11	transform_to_pixel_coordinates()	53
5.42.2	Variable Documentation	54
5.42.2.1	nogil	54
5.42.2.2	nopython	54
5.42.2.3	parallel	55
5.42.2.4	True	55
5.43	pyinsar.processing.geography.geodesy Namespace Reference	55
5.43.1	Function Documentation	55
5.43.1.1	compute_great_circle_distance_and_bearing()	56
5.43.1.2	compute_lonlat_from_distance_bearing()	56
5.43.1.3	compute_point_to_line_distance_for_array()	56
5.43.1.4	compute_point_to_line_distance_on_ellipsoid()	57
5.43.1.5	direct_vincenty_formula()	57
5.43.1.6	direct_vincenty_formula_for_array()	57
5.43.1.7	inverse_vincenty_formula()	57
5.43.1.8	inverse_vincenty_formula_for_array()	58
5.43.1.9	update_lambda()	58
5.43.2	Variable Documentation	58
5.43.2.1	nopython	58
5.44	pyinsar.processing.geography.geomorphometry Namespace Reference	58
5.44.1	Function Documentation	59
5.44.1.1	add_symmetric_border()	59

5.44.1.2	<code>compute_gradient_at_cell()</code>	59
5.44.1.3	<code>compute_horne_slope()</code>	59
5.44.2	Variable Documentation	59
5.44.2.1	<code>nopython</code>	59
5.45	<code>pyinsar.processing.instruments</code> Namespace Reference	60
5.46	<code>pyinsar.processing.instruments.sentinel</code> Namespace Reference	60
5.46.1	Function Documentation	61
5.46.1.1	<code>find_overlapping_valid_lines()</code>	61
5.46.1.2	<code>get_sentinel_extents()</code>	61
5.46.1.3	<code>get_valid_lines()</code>	62
5.46.1.4	<code>read_geolocation()</code>	62
5.46.1.5	<code>retrieve_azimuth_time()</code>	63
5.46.1.6	<code>select_valid_lines()</code>	63
5.46.1.7	<code>transform_slc()</code>	63
5.46.1.8	<code>update_geolocation_lines()</code>	64
5.47	<code>pyinsar.processing.isce</code> Namespace Reference	64
5.48	<code>pyinsar.processing.isce.input_file</code> Namespace Reference	64
5.48.1	Function Documentation	65
5.48.1.1	<code>create_product_xml()</code>	65
5.48.1.2	<code>create_topsApp_xml()</code>	65
5.48.1.3	<code>prepare_topsApps()</code>	66
5.49	<code>pyinsar.processing.machine_learning</code> Namespace Reference	66
5.50	<code>pyinsar.processing.machine_learning.geostatistics</code> Namespace Reference	67
5.51	<code>pyinsar.processing.machine_learning.geostatistics.direct_sampling</code> Namespace Reference	67
5.51.1	Function Documentation	68
5.51.1.1	<code>compute_continuous_distance()</code>	68
5.51.1.2	<code>compute_discrete_distance()</code>	68
5.51.1.3	<code>compute_neighborhood_lag_vectors()</code>	68

5.51.1.4	compute_neighborhoods()	69
5.51.1.5	find_closest_cell_in_training_image()	69
5.51.1.6	is_any_equal()	69
5.51.1.7	is_any_nan()	69
5.51.1.8	prepare_training_image()	70
5.51.1.9	run_ds()	70
5.51.1.10	run_parallel_ds()	70
5.51.1.11	simulate_ds_realization()	71
5.51.2	Variable Documentation	71
5.51.2.1	nopython	71
5.52	pyinsar.processing.machine_learning.geostatistics.geostatistics_utils Namespace Reference	73
5.52.1	Function Documentation	74
5.52.1.1	normalize()	74
5.52.1.2	standardize()	74
5.52.1.3	unflatten_index()	75
5.52.2	Variable Documentation	75
5.52.2.1	nogil	75
5.52.2.2	nopython	75
5.52.2.3	True	76
5.53	pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation Namespace Reference	76
5.53.1	Function Documentation	77
5.53.1.1	compute_averaged_cumulative_distribution_from_array()	77
5.53.1.2	compute_axis_aligned_ellipse_range()	77
5.53.1.3	compute_axis_aligned_neighborhood_shape()	77
5.53.1.4	compute_euclidean_distance()	77
5.53.1.5	compute_neighborhood_template()	78
5.53.1.6	get_data_to_data_matrix()	78

5.53.1.7	get_data_to_unknown_matrix()	78
5.53.1.8	get_neighborhood()	78
5.53.1.9	get_values_matrix()	79
5.53.1.10	inverse_standard_normal_cdf()	79
5.53.1.11	merge_secondary_data()	79
5.53.1.12	normal_score_tranform()	79
5.53.1.13	run_parallel_sgs()	80
5.53.1.14	run_sgs()	80
5.53.1.15	simulate_sgs_realization()	81
5.53.1.16	solve_kriging_system()	81
5.53.2	Variable Documentation	81
5.53.2.1	nogil	81
5.53.2.2	nopython	82
5.53.2.3	True	83
5.54	pyinsar.processing.machine_learning.geostatistics.variogram Namespace Reference	83
5.54.1	Function Documentation	84
5.54.1.1	compute_experimental_variogram()	84
5.54.1.2	compute_range_variogram()	85
5.54.1.3	compute_variogram()	85
5.54.1.4	exponential_variogram()	85
5.54.1.5	gaussian_variogram()	85
5.54.1.6	map_2D_variogram()	85
5.54.1.7	nugget_variogram()	86
5.54.1.8	spherical_variogram()	86
5.54.1.9	vectorized_exponential_variogram()	86
5.54.1.10	vectorized_gaussian_variogram()	86
5.54.1.11	vectorized_spherical_variogram()	87
5.54.2	Variable Documentation	87

5.54.2.1	nogil	87
5.54.2.2	nopython	87
5.54.2.3	True	89
5.55	pyinsar.processing.utilities Namespace Reference	89
5.56	pyinsar.processing.utilities.ann Namespace Reference	90
5.56.1	Function Documentation	90
5.56.1.1	buildCNN()	90
5.56.1.2	classify()	91
5.56.1.3	length_after_valid_window()	91
5.56.1.4	restoreGraph()	92
5.56.1.5	shuffleTrainingData()	92
5.56.1.6	train()	92
5.57	pyinsar.processing.utilities.deformations Namespace Reference	93
5.57.1	Function Documentation	93
5.57.1.1	calc_bounding_box()	93
5.57.1.2	determine_deformation_bounding_box()	94
5.57.1.3	determine_x_y_bounds()	94
5.58	pyinsar.processing.utilities.generic Namespace Reference	94
5.58.1	Function Documentation	95
5.58.1.1	coherence()	96
5.58.1.2	find_closest_time()	96
5.58.1.3	find_data_asf()	97
5.58.1.4	find_earthquake_pairs()	97
5.58.1.5	generateMatplotlibRectangle()	98
5.58.1.6	get_image_extents()	98
5.58.1.7	keypoints_align()	98
5.58.1.8	match_data()	99
5.58.1.9	phase_shift()	99

5.58.1.10 proj4StringToDictionary()	100
5.58.1.11 project_insar_data()	100
5.58.1.12 rotate()	101
5.58.1.13 scale_image()	101
5.58.1.14 select_max_matched_data()	101
5.58.1.15 sorted_alphanumeric()	102
5.58.1.16 subarray_slice()	102
5.58.1.17 translate()	103
5.59 pyinsar.processing.utilities.insar_simulator_utils Namespace Reference	103
5.59.1 Function Documentation	104
5.59.1.1 calc_bounding_box()	104
5.59.1.2 change_in_range_to_phase()	104
5.59.1.3 crop_array_from_center()	105
5.59.1.4 crop_nans()	105
5.59.1.5 determine_deformation_bounding_box()	105
5.59.1.6 determine_x_y_bounds()	106
5.59.1.7 generate_interferogram_from_deformation()	106
5.59.1.8 mask_deformation()	107
5.59.1.9 old_generate_interferogram_from_deformation()	108
5.59.1.10 phase_to_change_in_range()	108
5.59.1.11 retrieve_bounds()	109
5.59.1.12 wrap()	109
5.60 pyinsar.processing.utilities.machine_learning Namespace Reference	110
5.60.1 Function Documentation	110
5.60.1.1 divide_into_squares()	110
5.60.1.2 generate_minimum_ground_range_limits()	111
5.60.1.3 generate_phase_samples()	111
5.60.1.4 generate_phase_samples_from_looks_and_ranges()	112
5.60.1.5 rotate_image_list()	112

6	Class Documentation	115
6.1	pyinsar.processing.discovery.ClassifyCNN Class Reference	115
6.1.1	Detailed Description	116
6.1.2	Constructor & Destructor Documentation	116
6.1.2.1	__init__()	116
6.1.3	Member Function Documentation	116
6.1.3.1	process()	116
6.1.4	Member Data Documentation	117
6.1.4.1	batch_size	117
6.1.4.2	cnn_network_dir	117
6.1.4.3	compare_labels	117
6.1.4.4	config	117
6.1.4.5	size	117
6.1.4.6	stride	118
6.2	pyinsar.processing.discovery.Coherence Class Reference	118
6.2.1	Detailed Description	118
6.2.2	Constructor & Destructor Documentation	118
6.2.2.1	__init__()	118
6.2.3	Member Function Documentation	119
6.2.3.1	process()	119
6.2.4	Member Data Documentation	119
6.2.4.1	pairing	119
6.2.4.2	use_progress_bar	119
6.2.4.3	window	120
6.3	pyinsar.processing.discovery.Coregister Class Reference	120
6.3.1	Detailed Description	120
6.3.2	Constructor & Destructor Documentation	120
6.3.2.1	__init__()	120

6.3.3	Member Function Documentation	121
6.3.3.1	process()	121
6.4	pyinsar.processing.data_fetcher.okada.DataFetcher Class Reference	121
6.4.1	Detailed Description	122
6.4.2	Constructor & Destructor Documentation	122
6.4.2.1	__init__()	122
6.4.3	Member Function Documentation	122
6.4.3.1	output()	122
6.5	pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever Class Reference	123
6.5.1	Detailed Description	123
6.5.2	Constructor & Destructor Documentation	123
6.5.2.1	__init__()	123
6.5.3	Member Function Documentation	124
6.5.3.1	output()	124
6.5.3.2	perturb()	124
6.6	pyinsar.processing.utilities.machine_learning.DataRetriever Class Reference	124
6.6.1	Detailed Description	125
6.6.2	Constructor & Destructor Documentation	125
6.6.2.1	__init__()	125
6.6.3	Member Function Documentation	126
6.6.3.1	get_images()	126
6.6.3.2	get_num_images()	126
6.6.4	Member Data Documentation	126
6.6.4.1	chunk_size	126
6.6.4.2	data_file_dict	127
6.6.4.3	dtype	127
6.6.4.4	label_list	127
6.6.4.5	size	127

6.7	pyinsar.processing.discovery.deburst.Deburst Class Reference	127
6.7.1	Detailed Description	128
6.7.2	Constructor & Destructor Documentation	128
6.7.2.1	<code>__init__()</code>	128
6.7.3	Member Function Documentation	128
6.7.3.1	<code>process()</code>	128
6.8	pyinsar.processing.discovery.DeformationToPhase Class Reference	129
6.8.1	Detailed Description	129
6.8.2	Constructor & Destructor Documentation	129
6.8.2.1	<code>__init__()</code>	129
6.8.3	Member Function Documentation	130
6.8.3.1	<code>process()</code>	130
6.9	pyinsar.processing.deformation.elastic_halfspace.fault.Fault Class Reference	130
6.9.1	Detailed Description	131
6.9.2	Constructor & Destructor Documentation	131
6.9.2.1	<code>__init__()</code>	131
6.9.3	Member Function Documentation	132
6.9.3.1	<code>generateDeformation()</code>	132
6.9.4	Member Data Documentation	132
6.9.4.1	<code>cell_centroids</code>	133
6.9.4.2	<code>cell_length</code>	133
6.9.4.3	<code>cell_width</code>	133
6.9.4.4	<code>depth</code>	133
6.9.4.5	<code>dip</code>	133
6.9.4.6	<code>length</code>	133
6.9.4.7	<code>poisson_ratio</code>	133
6.9.4.8	<code>strike</code>	134
6.9.4.9	<code>unrotated_x</code>	134

6.9.4.10	unrotated_y	134
6.9.4.11	width	134
6.9.4.12	x_center	134
6.9.4.13	y_center	134
6.10	pyinsar.processing.utilities.generic.FindNearestPixel Class Reference	135
6.10.1	Detailed Description	135
6.10.2	Constructor & Destructor Documentation	135
6.10.2.1	__init__()	135
6.10.3	Member Function Documentation	136
6.10.3.1	__call__()	136
6.11	pyinsar.processing.discovery.FlatEarth Class Reference	136
6.11.1	Detailed Description	137
6.11.2	Constructor & Destructor Documentation	137
6.11.2.1	__init__()	137
6.11.3	Member Function Documentation	137
6.11.3.1	process()	137
6.11.4	Member Data Documentation	139
6.11.4.1	k	139
6.12	pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher Class Reference	139
6.12.1	Detailed Description	139
6.12.2	Constructor & Destructor Documentation	140
6.12.2.1	__init__()	140
6.12.3	Member Function Documentation	140
6.12.3.1	output()	140
6.13	pyinsar.processing.discovery.interferogram.Interferogram Class Reference	141
6.13.1	Detailed Description	141
6.13.2	Constructor & Destructor Documentation	141
6.13.2.1	__init__()	141

6.13.3	Member Function Documentation	142
6.13.3.1	process()	142
6.14	pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod Class Reference	142
6.14.1	Member Data Documentation	142
6.14.1.1	nopython	143
6.14.1.2	ORDINARY	143
6.14.1.3	SIMPLE	143
6.15	pyinsar.processing.discovery.LOS_Deformation Class Reference	143
6.15.1	Detailed Description	143
6.16	pyinsar.processing.utilities.generic.OrbitInterpolation Class Reference	144
6.16.1	Detailed Description	144
6.16.2	Constructor & Destructor Documentation	144
6.16.2.1	__init__()	144
6.16.3	Member Function Documentation	145
6.16.3.1	__call__()	145
6.16.3.2	get_start_date()	145
6.17	pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType Class Reference	146
6.17.1	Member Data Documentation	146
6.17.1.1	LINEAR	146
6.17.1.2	RANDOM	146
6.18	pyinsar.processing.instruments.sentinel.RampPolynomial Class Reference	146
6.18.1	Detailed Description	147
6.18.2	Constructor & Destructor Documentation	147
6.18.2.1	__init__()	147
6.18.3	Member Function Documentation	147
6.18.3.1	__call__()	147
6.19	pyinsar.processing.discovery.RotateSquares Class Reference	148

6.19.1 Detailed Description	148
6.19.2 Constructor & Destructor Documentation	148
6.19.2.1 <code>__init__()</code>	148
6.19.3 Member Function Documentation	149
6.19.3.1 <code>process()</code>	149
6.20 <code>pyinsar.processing.instruments.sentinel.SentinelRamp</code> Class Reference	149
6.20.1 Detailed Description	150
6.20.2 Constructor & Destructor Documentation	150
6.20.2.1 <code>__init__()</code>	150
6.20.3 Member Function Documentation	150
6.20.3.1 <code>__call__()</code>	151
6.20.4 Member Data Documentation	151
6.20.4.1 <code>modulation</code>	151
6.21 <code>pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses</code> Class Reference	151
6.21.1 Detailed Description	152
6.21.2 Constructor & Destructor Documentation	152
6.21.2.1 <code>__init__()</code>	152
6.21.3 Member Function Documentation	152
6.21.3.1 <code>process()</code>	153
6.21.4 Member Data Documentation	153
6.21.4.1 <code>class_name</code>	153
6.21.4.2 <code>colors</code>	153
6.22 <code>pyinsar.processing.discovery.TemporalDecorrelation</code> Class Reference	153
6.22.1 Detailed Description	154
6.22.2 Constructor & Destructor Documentation	154
6.22.2.1 <code>__init__()</code>	154
6.22.3 Member Function Documentation	155
6.22.3.1 <code>process()</code>	155

6.23	pyinsar.processing.discovery.TrainCNN Class Reference	155
6.23.1	Detailed Description	156
6.23.2	Constructor & Destructor Documentation	156
6.23.2.1	<code>__init__()</code>	156
6.23.3	Member Function Documentation	156
6.23.3.1	<code>process()</code>	156
6.23.4	Member Data Documentation	157
6.23.4.1	<code>batch_size</code>	157
6.23.4.2	<code>cnn_network_dir</code>	157
6.23.4.3	<code>config</code>	157
6.24	pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType Class Reference	157
6.24.1	Member Data Documentation	158
6.24.1.1	CONTINUOUS	158
6.24.1.2	DISCRETE	158
6.25	pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel Class Reference	158
6.25.1	Detailed Description	158
6.25.2	Member Data Documentation	159
6.25.2.1	EXPONENTIAL	159
6.25.2.2	GAUSSIAN	159
6.25.2.3	NUGGET	159
6.25.2.4	SPHERICAL	159
6.26	pyinsar.processing.discovery.WrapPhase Class Reference	159
6.26.1	Detailed Description	160
6.26.2	Member Function Documentation	160
6.26.2.1	<code>process()</code>	160

7 File Documentation	161
7.1 data_import/import_georaster.py File Reference	161
7.2 data_import/import_raster.py File Reference	161
7.3 data_import/import_srcmod.py File Reference	162
7.4 data_import/import_utils.py File Reference	162
7.5 data_import/uavsar.py File Reference	162
7.6 output/export_georaster.py File Reference	163
7.7 output/plot_raster.py File Reference	163
7.8 processing/corrections/topography.py File Reference	163
7.9 processing/corrections/troposphere.py File Reference	164
7.10 processing/data_fetcher/gdal.py File Reference	164
7.11 processing/data_fetcher/hdf_retriever.py File Reference	164
7.12 processing/deformation/elastic_halfspace/fault.py File Reference	165
7.13 processing/deformation/elastic_halfspace/mogi.py File Reference	165
7.14 processing/deformation/elastic_halfspace/okada.py File Reference	165
7.15 processing/data_fetcher/okada.py File Reference	167
7.16 processing/deformation/elastic_halfspace/pipe.py File Reference	167
7.17 processing/deformation/elastic_halfspace/surface_load.py File Reference	168
7.18 processing/discovery/classify_cnn.py File Reference	168
7.19 processing/discovery/coherence.py File Reference	168
7.20 processing/discovery/coregister.py File Reference	168
7.21 processing/discovery/deburst.py File Reference	169
7.22 processing/discovery/deformation_to_phase.py File Reference	169
7.23 processing/discovery/flat_earth.py File Reference	169
7.24 processing/discovery/interferogram.py File Reference	170
7.25 processing/discovery/los_deformation.py File Reference	170
7.26 processing/discovery/rotate_squares.py File Reference	170
7.27 processing/discovery/shown_cnn_classes.py File Reference	171

7.28 processing/discovery/temporal_decorrelation.py File Reference	171
7.29 processing/discovery/train_cnn.py File Reference	171
7.30 processing/discovery/wrap_phase.py File Reference	171
7.31 processing/geography/coordinates.py File Reference	172
7.32 processing/geography/geodesy.py File Reference	173
7.33 processing/geography/geomorphometry.py File Reference	173
7.34 processing/instruments/sentinel.py File Reference	174
7.35 data_import/sentinel.py File Reference	174
7.36 processing/isce/input_file.py File Reference	175
7.37 processing/machine_learning/geostatistics/direct_sampling.py File Reference	175
7.38 processing/machine_learning/geostatistics/geostatistics_utils.py File Reference	176
7.39 processing/machine_learning/geostatistics/sequential_gaussian_simulation.py File Reference	177
7.40 processing/machine_learning/geostatistics/variogram.py File Reference	178
7.41 processing/utilities/ann.py File Reference	179
7.42 processing/utilities/deformations.py File Reference	180
7.43 processing/utilities/generic.py File Reference	180
7.44 processing/utilities/insar_simulator_utils.py File Reference	181
7.45 processing/utilities/machine_learning.py File Reference	182

Chapter 1

Namespace Index

1.1 Packages

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

pyinsar	9
pyinsar.data_import	9
pyinsar.data_import.import_georaster	9
pyinsar.data_import.import_raster	11
pyinsar.data_import.import_srcmod	13
pyinsar.data_import.import_utils	14
pyinsar.data_import.sentinel	15
pyinsar.data_import.uavsar	17
pyinsar.output	17
pyinsar.output.export_georaster	17
pyinsar.output.plot_raster	18
pyinsar.processing	20
pyinsar.processing.corrections	20
pyinsar.processing.corrections.topography	20
pyinsar.processing.corrections.troposphere	21
pyinsar.processing.data_fetcher	22
pyinsar.processing.data_fetcher.gdal	23
pyinsar.processing.data_fetcher.hdf_retriever	23
pyinsar.processing.data_fetcher.okada	23
pyinsar.processing.deformation	23
pyinsar.processing.deformation.elastic_halfspace	23
pyinsar.processing.deformation.elastic_halfspace.fault	24
pyinsar.processing.deformation.elastic_halfspace.mogi	24
pyinsar.processing.deformation.elastic_halfspace.okada	24
pyinsar.processing.deformation.elastic_halfspace.pipe	42
pyinsar.processing.deformation.elastic_halfspace.surface_load	43
pyinsar.processing.discovery	44
pyinsar.processing.discovery.classify_cnn	45
pyinsar.processing.discovery.coherence	45
pyinsar.processing.discovery.coregister	45
pyinsar.processing.discovery.deburst	45

pyinsar.processing.discovery.deformation_to_phase	46
pyinsar.processing.discovery.flat_earth	46
pyinsar.processing.discovery.interferogram	46
pyinsar.processing.discovery.los_deformation	46
pyinsar.processing.discovery.rotate_squares	46
pyinsar.processing.discovery.shown_cnn_classes	47
pyinsar.processing.discovery.temporal_decorrelation	47
pyinsar.processing.discovery.train_cnn	47
pyinsar.processing.discovery.wrap_phase	48
pyinsar.processing.geography	48
pyinsar.processing.geography.coordinates	48
pyinsar.processing.geography.geodesy	55
pyinsar.processing.geography.geomorphometry	58
pyinsar.processing.instruments	60
pyinsar.processing.instruments.sentinel	60
pyinsar.processing.isce	64
pyinsar.processing.isce.input_file	64
pyinsar.processing.machine_learning	66
pyinsar.processing.machine_learning.geostatistics	67
pyinsar.processing.machine_learning.geostatistics.direct_sampling	67
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils	73
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation	76
pyinsar.processing.machine_learning.geostatistics.variogram	83
pyinsar.processing.utilities	89
pyinsar.processing.utilities.ann	90
pyinsar.processing.utilities.deformations	93
pyinsar.processing.utilities.generic	94
pyinsar.processing.utilities.insar_simulator_utils	103
pyinsar.processing.utilities.machine_learning	110

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

object	
pyinsar.processing.deformation.elastic_halfspace.fault.Fault	130
pyinsar.processing.instruments.sentinel.RampPolynomial	146
pyinsar.processing.instruments.sentinel.SentinelRamp	149
pyinsar.processing.utilities.generic.FindNearestPixel	135
pyinsar.processing.utilities.generic.OrbitInterpolation	144
pyinsar.processing.utilities.machine_learning.DataRetriever	124
DataFetcherBase	
pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher	139
pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever	123
pyinsar.processing.data_fetcher.okada.DataFetcher	121
Enum	
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType	146
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType	157
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod	142
pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel	158
PipelineItem	
pyinsar.processing.discovery.ClassifyCNN	115
pyinsar.processing.discovery.Coherence	118
pyinsar.processing.discovery.Coregister	120
pyinsar.processing.discovery.deburst.Deburst	127
pyinsar.processing.discovery.DeformationToPhase	129
pyinsar.processing.discovery.FlatEarth	136
pyinsar.processing.discovery.interferogram.Interferogram	141
pyinsar.processing.discovery.LOS_Deformation	143
pyinsar.processing.discovery.RotateSquares	148
pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses	151
pyinsar.processing.discovery.TemporalDecorrelation	153
pyinsar.processing.discovery.TrainCNN	155
pyinsar.processing.discovery.WrapPhase	159

Chapter 3

Class Index

3.1 Class List

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

pyinsar.processing.discovery.ClassifyCNN	
Train a CNN	115
pyinsar.processing.discovery.Coherence	
Calculate coherence between single-look complex SAR images	118
pyinsar.processing.discovery.Coregister	
*** In Development *** Pipeline item to coregister images	120
pyinsar.processing.data_fetcher.okada.DataFetcher	
Generates data from an Okada model	121
pyinsar.processing.data_fetcher.hdf_retriever.DataRetriever	
Data fetcher for retrieving hdf image data made for training in convolutional neural networks	123
pyinsar.processing.utilities.machine_learning.DataRetriever	
Class for retrieving data from an hdf file	124
pyinsar.processing.discovery.deburst.Deburst	
Debursts Sentinel-1 TOPSAR data	127
pyinsar.processing.discovery.DeformationToPhase	
Convert deformation to phas	129
pyinsar.processing.deformation.elastic_halfspace.fault.Fault	
*** In Development *** Model a fault as a collection of small okada faults	130
pyinsar.processing.utilities.generic.FindNearestPixel	
Find the nearest given a time	135
pyinsar.processing.discovery.FlatEarth	
*** In Development *** Remove flat Earth contribution from interferogram	136
pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher	
Data fetcher for loading Images produced compatible with GDAL	139
pyinsar.processing.discovery.interferogram.Interferogram	
Create Interferogram from SLC data	141
pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.KrigingMethod	
.	142
pyinsar.processing.discovery.LOS_Deformation	
*** In Development ***	143
pyinsar.processing.utilities.generic.OrbitInterpolation	
Class for interpolating satellite positions	144

pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.PathType	146
pyinsar.processing.instruments.sentinel.RampPolynomial	
Polynomial used for quantities relating to deramping sentinel	146
pyinsar.processing.discovery.RotateSquares	
Generate new images by rotating subsections of data defined by Shapely squares	148
pyinsar.processing.instruments.sentinel.SentinelRamp	
Calculate the combined ramp and modulated phase in Sentinel	149
pyinsar.processing.discovery.shown_cnn_classes.ShowCNNClasses	
Display CNN Classifications on segments of an image	151
pyinsar.processing.discovery.TemporalDecorrelation	
Pipeline item to add temporal decorrelation to some phase	153
pyinsar.processing.discovery.TrainCNN	
Train a CNN	155
pyinsar.processing.machine_learning.geostatistics.geostatistics_utils.VariableType	157
pyinsar.processing.machine_learning.geostatistics.variogram.VariogramModel	
2D theoretical variogram	158
pyinsar.processing.discovery.WrapPhase	
Pipeline Item that wraps phase	159

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

data_import/import_georaster.py	161
data_import/import_raster.py	161
data_import/import_srcmod.py	162
data_import/import_utils.py	162
data_import/sentinel.py	174
data_import/uavsar.py	162
output/export_georaster.py	163
output/plot_raster.py	163
processing/corrections/topography.py	163
processing/corrections/troposphere.py	164
processing/data_fetcher/gdal.py	164
processing/data_fetcher/hdf_retriever.py	164
processing/data_fetcher/okada.py	167
processing/deformation/elastic_halfspace/fault.py	165
processing/deformation/elastic_halfspace/mogi.py	165
processing/deformation/elastic_halfspace/okada.py	165
processing/deformation/elastic_halfspace/pipe.py	167
processing/deformation/elastic_halfspace/surface_load.py	168
processing/discovery/classify_cnn.py	168
processing/discovery/coherence.py	168
processing/discovery/coregister.py	168
processing/discovery/deburst.py	169
processing/discovery/deformation_to_phase.py	169
processing/discovery/flat_earth.py	169
processing/discovery/interferogram.py	170
processing/discovery/los_deformation.py	170
processing/discovery/rotate_squares.py	170
processing/discovery/shown_cnn_classes.py	171
processing/discovery/temporal_decorrelation.py	171
processing/discovery/train_cnn.py	171
processing/discovery/wrap_phase.py	171

processing/geography/ coordinates.py	172
processing/geography/ geodesy.py	173
processing/geography/ geomorphometry.py	173
processing/instruments/ sentinel.py	174
processing/isce/ input_file.py	175
processing/machine_learning/geostatistics/ direct_sampling.py	175
processing/machine_learning/geostatistics/ geostatistics_utils.py	176
processing/machine_learning/geostatistics/ sequential_gaussian_simulation.py	177
processing/machine_learning/geostatistics/ variogram.py	178
processing/utilities/ ann.py	179
processing/utilities/ deformations.py	180
processing/utilities/ generic.py	180
processing/utilities/ insar_simulator_utils.py	181
processing/utilities/ machine_learning.py	182

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

<i>gdal_georaster</i>	A georaster opened with GDAL
<i>remove_ndv</i>	Replace the no-data value as mentionned in the label by np.nan
<i>as_float</i>	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

<i>gdal_georaster</i>	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

<i>georaster_path</i>	Location of the georaster
<i>read_only</i>	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

<i>gdal_georaster</i>	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

<code>tropodelay_header_path</code>	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

<code>file_url</code>	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

<i>srcmod_data</i>	src mod data read in from the .mat file
<i>dtype</i>	Data type to use
<i>skip_sanity_check</i>	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

<i>url</i>	The URL where the file is
<i>folder_path</i>	Path to the folder where the downloaded file will be stored
<i>username</i>	username for authentication, if needed
<i>password</i>	Password for authentication, if needed
<i>filename</i>	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

<i>product_folder</i>	The folder where the Sentinel-1 products are
<i>orbit_folder</i>	The folder where to put the orbit files
<i>username</i>	The username for authentication on Earthdata
<i>password</i>	The password for authentication 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

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

<i>in_satellite_file</i>	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

<i>in_file</i>	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

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

<i>azimuth_time</i>	Pandas time series data conatining the time of each azimuth line
<i>latlon</i>	Function to compute latitude and longitude for each pixel coordinate
<i>orbit_interp</i>	Function to compute satellite positions
<i>start_x</i>	Starting x pixel
<i>end_x</i>	Ending pixel x pxiel
<i>start_y</i>	Starting y pixel
<i>end_y</i>	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 compatible 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 [l1](#) (xi, eta, q, delta, nu, R, X, d_tild)

Okada's surface displacement.
- def [l2](#) (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def [l3](#) (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def [l4](#) (xi, eta, q, delta, nu, R, d_tild)
- def [l5](#) (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 [l1_int](#) (xi, eta, z, y, delta, c, d, q, R)
Okada's internal displacement.
- def [l2_int](#) (xi, eta, z, y, delta, c, d, q, R)
- def [l3_int](#) (xi, eta, z, y, delta, c, d, q, R)
- def [l4_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

<i>closed_pipe_x</i>	x coordinate for the pipe's center
<i>closed_pipe_y</i>	y coordinate for the pipe's center
<i>closed_pipe_depth_1</i>	Pipe's top depth
<i>closed_pipe_depth_2</i>	Pipe's bottom depth
<i>closed_pipe_radius</i>	Pipe's radius
<i>poisson_ratio</i>	Poisson's ratio
<i>pressurization</i>	Change of pressure applied to the pipe
<i>shear_modulus</i>	Shear modulus
<i>xx_array</i>	x coordinate for the domain within a 2D array
<i>yy_array</i>	y coordinate 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

<i>open_pipe_x</i>	x cooordinate for the pipe's center
<i>open_pipe_y</i>	y cooordinate for the pipe's center
<i>open_pipe_depth_0</i>	Pipe's top depth with minimal pressurization
<i>open_pipe_depth_1</i>	Pipe's top depth with maximal pressurization
<i>open_pipe_depth_2</i>	Pipe's bottom depth
<i>open_pipe_radius</i>	Pipe's radius
<i>poisson_ratio</i>	Poisson's ratio
<i>pressurization</i>	Change of pressure applied to the pipe
<i>shear_modulus</i>	Shear modulus
<i>xx_array</i>	x cooordinate for the domain within a 2D array
<i>yy_array</i>	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

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

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

5.37 pyinsar.processing.discovery.shown_cnn_classes Namespace Reference

Classes

- class [ShowCNNClasses](#)
Display 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_subgeoraster](#) (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

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

<i>longitude</i>	A float for the longitude
<i>latitude</i>	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

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

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

<i>array</i>	A 2D NumPy array
<i>subarray_shape</i>	The shape of the sub-arrays
<i>steps</i>	The step between each sub-array for each axis, to avoid sampling all the possible sub-arrays
<i>is_shape_centered</i>	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

<i>array</i>	A 3D NumPy array. The first dimension represents the variables, the other two the x and y axis.
<i>subarray_shape</i>	The 2D shape of the sub-arrays
<i>steps</i>	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

<i>array</i>	The array
<i>border_size</i>	The size of the border

Returns

The expended array

Parameters

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

Returns

The gradient value for the cell

Parameters

<i>array</i>	The array
<i>grid_yx_spacing</i>	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](#)
Calculate 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

<i>metadata_tree</i>	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

<i>geolocation</i>	Geolocation data read in by read_geolocation
<i>offset</i>	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

<i>metadata_tree</i>	Sentinel XML metadata tree
<i>per_burst</i>	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

<i>tree</i>	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

<i>in_tree</i>	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

<i>data</i>	Input SLC data
<i>tree</i>	Metadata as an ElementTree
<i>cut</i>	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

<i>slc</i>	Input slc
<i>deramped_phase</i>	Phase to be removed before the transformation and to be readed afterwards
<i>transformation_matrix</i>	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

<i>tree</i>	Sentinel XML metadata
<i>azimuth_times</i>	Azimuth times
<i>geolocation_data</i>	Geolocation data read in by <code>read_geolocation</code>

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

<i>xml_path</i>	Path to the xml file
<i>product_path</i>	Path to the Sentinel-1 product
<i>product_type</i>	Master or slave product
<i>product_output_path</i>	Path for the processing outputs of this product
<i>product_orbit_path</i>	Path to the folder containing orbit files
<i>product_auxiliary_data_path</i>	Path to the folder containing auxiliary data
<i>do_add</i>	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

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

<i>neighborhood_shape</i>	The maximal coverage of the neighborhood along each axis
<i>grid_yx_spacing</i>	The cell size along each axis (y, x)
<i>delta</i>	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

<i>training_image_array</i>	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
<i>ti_j</i>	Index along the y axis of the initial cell to visit in the training image
<i>ti_i</i>	Index along the x axis of the initial cell to visit in the training image
<i>ti_ranges_max</i>	Squared difference between the min and max value of each variable
<i>neighbor_indexes</i>	Indexes of the neighborhood from the cell to simulate
<i>neighbor_values</i>	Values of the neighborhood in the simulation grid
<i>neighbor_numbers</i>	Number of neighbors for each variable
<i>min_distances</i>	The minimal distance of each variable found so far
<i>var_k</i>	Index of the variable
<i>max_non_matching_proportion</i>	Authorized proportion of non-matching nodes, i.e., whose distance is below the threshold for the variable
<i>no_data_value</i>	The no-data value, which defines the cell to simulate

Returns

The distance

Parameters

<i>training_image_array</i>	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
<i>ti_j</i>	Index along the y axis of the initial cell to visit in the training image
<i>ti_i</i>	Index along the x axis of the initial cell to visit in the training image

Parameters

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

Returns

The distance

Parameters

<i>list↔ _1</i>	The list
<i>value</i>	The value

Returns

True if there is the value, false otherwise

Parameters

<i>list↔ _1</i>	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

<i>x</i>	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

<i>x</i>	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

<i>flattened_index</i>	The flattened index (i.e., a single integer)
<i>array_shape</i>	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 )
```

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

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

Compute the inverse of a normal cumulative distribution

```
@param x: A float or array

@return The values normally distributed
```

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

Merge several secondary data (see Babak and Deutsch, 2009, doi:10.1016/j.petro.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
```

```
def pyinsar.processing.machine_learning.geostatistics.sequential_gaussian_simulation.normal_↵
score_transform (
    value_array )
```

Transform the values of an array to a normal distribution.

Parameters

<i>value_array</i>	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

<i>neighborhood_range</i>	The major and minor axis length of the ellipse
<i>neighborhood_azimuth_rad</i>	The azimuth giving the orientation of the major axis

Returns

The extent

Parameters

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

Returns

The shape

Parameters

<i>neighborhood</i>	The data around the cell to estimate
<i>simulation_array</i>	The simulation grid

Returns

The matrix

Parameters

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

Returns

The matrix

Parameters

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

<i>distance</i>	The distance between the two points
<i>vario_range</i>	The variogram range
<i>variance_contribution</i>	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

<i>distance</i>	The distance between the two points
<i>vario_range</i>	The variogram range
<i>variance_contribution</i>	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

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

Returns

The value of the variogram

Parameters

<i>delta_y</i>	The distance between the two points along the y axis
<i>delta_x</i>	The distance between the two points along the x axis
<i>vario_models</i>	The models for the variogram
<i>vario_sills</i>	The sills for the variogram
<i>vario_ranges</i>	The major and minor ranges for the variogram
<i>rotation_matrix</i>	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

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

<i>image_height</i>	Height of image in pixels
<i>image_width</i>	Width of image in pixels
<i>model_dir</i>	Directory to save network too
<i>rate</i>	Learning rate
<i>config</i>	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

<i>image_data</i>	Input data
<i>model_dir</i>	Directory where network is stored
<i>batch_size</i>	Batch size to use for classifying data
<i>config</i>	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

<i>length</i>	Initial length
<i>window</i>	Size of convolution window
<i>stride</i>	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

<i>model_dir</i>	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

<i>data</i>	Input data
<i>labels</i>	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

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

<i>image</i>	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

<i>deformations</i>	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

<i>deformations</i>	Input deformations
<i>x_array</i>	X coordinates
<i>y_array</i>	Y coordinates
<i>offset</i>	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

<i>s1</i>	The first single look complex image
<i>s2</i>	The second single look complex image
<i>window</i>	Tuple specifying y, and x window size
<i>topo_phase</i>	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

<i>time</i>	Pandas series of datetimes
<i>date</i>	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

<i>lat</i>	Latitude
<i>lon</i>	Longitude
<i>processingLevel</i>	Processing level of data
<i>platform</i>	Instrument to search
<i>kwargs</i>	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

<i>organized_data</i>	Dictionary of information about data that has been organized into overlapping images
<i>date</i>	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

<i>extent</i>	Container holding the extent (x_min, x_max, y_min, y_max)
<i>kwargs</i>	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

<i>geotransform</i>	Geo transform for converting between pixel and projected coordinates
<i>shape</i>	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

<i>img1</i>	First image
<i>img2</i>	Second image
<i>max_matches</i>	Maximum number of matches between the two images
<i>invert</i>	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

<i>sentinel_data_list</i>	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

<i>data</i>	Input data
<i>phase</i>	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

<i>proj4_string</i>	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

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

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

<i>sentinel_data_list</i>	
---------------------------	--

Returns

:

5.58.1.15 sorted_alphanumeric()

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

Sort a list of strings with numbers.

Parameters

/	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

<i>index</i>	Which chunk to select
<i>num_items</i>	Number of items in a chunk

Returns

A slice for selecting $\text{index} * \text{num_items}$ to $(\text{index} + 1) * \text{num_items}$

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

<i>col_vectors</i>	Array of column vectors
<i>delta_x</i>	Move this many units in the x direction
<i>delta_y</i>	Move this many units in the y direction
<i>delta_z</i>	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)
Calculate the bounding box around an image using the li threshold.
- def [retrieve_bounds](#) (thresh_image)
Retrieve the bounds of an image that has been thresholded.
- 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 )
```

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

Parameters

<i>image</i>	Input image
<i>threshold_function</i>	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

<i>los_deformation</i>	Change in distance along line of site
<i>wavelength</i>	Wavelength of radar
<i>k</i>	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

<i>array</i>	The array
<i>crop_shape</i>	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

<i>image</i>	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

<i>deformations</i>	Input deformations
<i>largest_box</i>	Choose a bounding max that encompasses all selected values in all dimensions
<i>kwargs</i>	Any additional keyword arguments passed to <code>calc_bounding_box</code>

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

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

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

<i>track_angle</i>	Satellite track angle
<i>min_ground_range_1</i>	Minimum ground range to deformations for first pass
<i>height_1</i>	Height of satellite for first pass
<i>is_right_looking</i>	The satellite is looking to the right
<i>wavelength</i>	Wavelength of the signal
<i>k</i>	number of passes (1 or 2)
<i>deformation</i>	map of deformation
<i>xx</i>	x coordinates of deformation
<i>yy</i>	y coordinates of deformation
<i>projected_topography</i>	Elevation data
<i>min_ground_range_2</i>	Minimum ground range to deformations for second pass
<i>height_2</i>	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

<i>deformation</i>	Deformation to mask
<i>threshold_function</i>	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

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

<i>phase</i>	Input phase
<i>wavelength</i>	Wavelength of radar
<i>k</i>	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 thresholded.

Parameters

<i>thresh_image</i>	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

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

Returns

The wrapped array (in radian between $-\pi$ and π)

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 ground 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

<i>image</i>	Source image
<i>size</i>	Size of one side of the square patch
<i>stride</i>	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

<i>satellite_height</i>	Height of satellite
<i>incidence_ranges</i>	Range of valid incidence angles (shape of [:,2])
<i>image_size</i>	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

<i>deformation</i>	Array containing deformation
<i>satellite_height</i>	Height of Satellite
<i>radar_wavelength</i>	Wavelength of radar
<i>cell_size</i>	Size of cell (length of one side of the cell)
<i>image_size</i>	Ignored?
<i>stride</i>	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 ground ranges.

Parameters

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

Returns

array containing parameters 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

<i>in_image_extents</i>	List of the extents of the images being rotated
<i>in_image_list</i>	List of images to rotate
<i>progress</i>	Show a progress bar

Returns

array of image extents, and array of rotated images

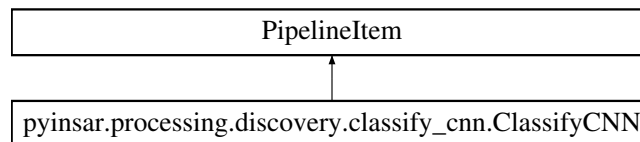
Chapter 6

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

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

<i>obj_data</i>	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`

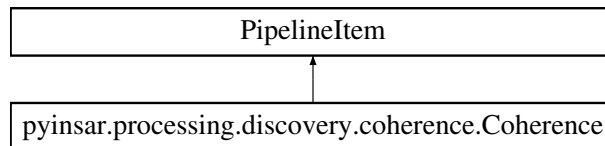
The documentation for this class was generated from the following file:

- [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

<i>str_description</i>	Short string describing item
<i>window</i>	Tuple indicating the y and x window size
<i>pairing</i>	How to pair slc images. "neighbor" computes coherence between neighboring images
<i>use_progress_bar</i>	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

<i>obj_data</i>	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`

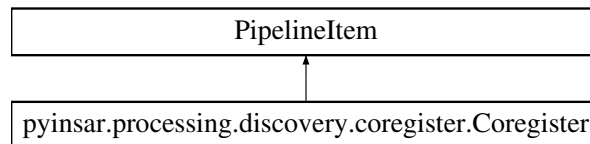
The documentation for this class was generated from the following file:

- `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

<i>str_description</i>	String describing item
<i>ap_paramList[reg_type]</i>	Registration method (currently supports 'imreg_translation', 'imreg_affine' and 'keypoints')
<i>image_limits</i>	Limits of image to use when comparing for coregistration
<i>num_iterations</i>	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

<i>obj_data</i>	Image data wrapper
-----------------	--------------------

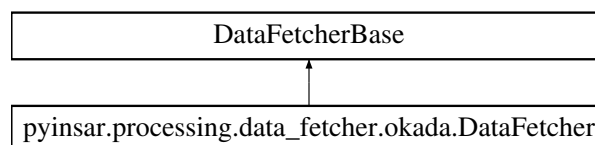
The documentation for this class was generated from the following file:

- 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

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

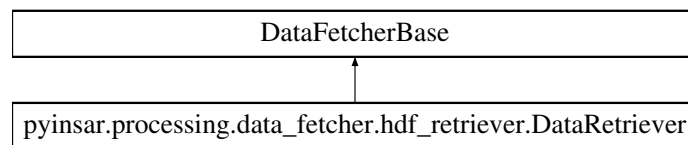
The documentation for this class was generated from the following file:

- `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

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

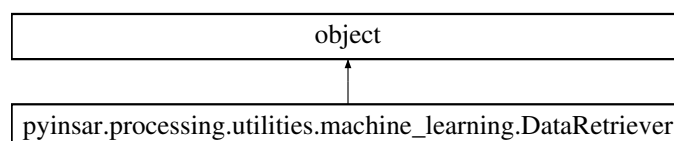
The documentation for this class was generated from the following file:

- [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

<i>file_name_list</i>	List of hdf filenames
<i>label_list</i>	List of labels
<i>size</i>	Tuple containing the size of the images
<i>dtype</i>	Data type of images
<i>chunk_size</i>	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

<i>index</i>	Array with shape <code>[:,2]</code> , 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
```

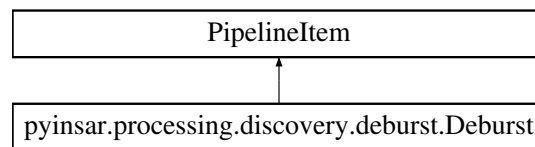
The documentation for this class was generated from the following file:

- 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

<i>str_description</i>	String description of item
<i>cut_on_master</i>	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

<i>obj_data</i>	Data wrapper
-----------------	--------------

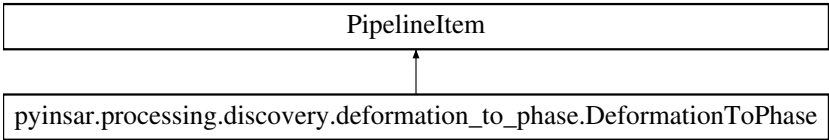
The documentation for this class was generated from the following file:

- `processing/discovery/debust.py`

6.8 pyinsar.processing.discovery.DeformationToPhase Class Reference

Convert deformation to phas.

Inheritance diagram for pyinsar.processing.discovery.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

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

<i>obj_data</i>	Image data wrapper
-----------------	--------------------

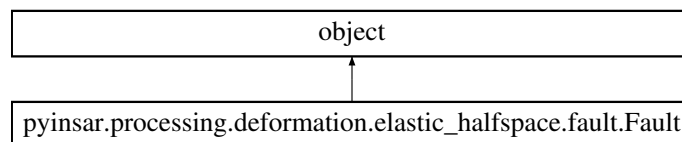
The documentation for this class was generated from the following file:

- 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:



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

<i>x_center</i>	x centroid of fault
<i>y_center</i>	y centroid of fault
<i>depth</i>	Depth to centroid of fault

Parameters

<i>length</i>	Length of fault (along strike)
<i>width</i>	Width of fault (along dip)
<i>strike</i>	Angle from north of the fault direction
<i>dip</i>	Dip angle
<i>num_elements_length</i>	Number of elements in the length direction
<i>num_elements_width</i>	Number of elements in the width direction
<i>poisson_ratio</i>	Poisson ratio
<i>dtype</i>	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

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

Returns

Surface deformations at specified 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`

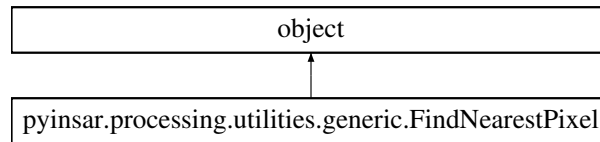
The documentation for this class was generated from the following file:

- [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

<i>aztime</i>	Input azimuth time series
<i>start_date</i>	The starting date to use when computing 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

<code>in_time</code>	Input time
----------------------	------------

Returns

: Pixel that is closest to the input time

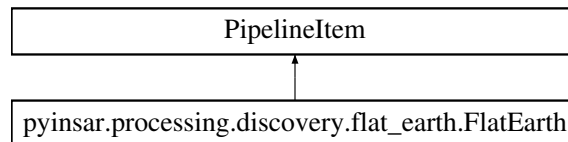
The documentation for this class was generated from the following file:

- [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

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

<code>obj_data</code>	Input image data wrapper
-----------------------	--------------------------

6.11.4 Member Data Documentation

6.11.4.1 k

`pyinsar.processing.discovery.FlatEarth.k`

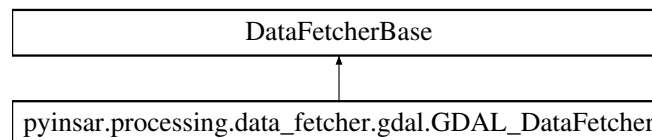
The documentation for this class was generated from the following file:

- `processing/discovery/flat_earth.py`

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

Data fetcher for loading Images produced compatible 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 compatible 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

<i>filename_list</i>	List of filenames of ISCE interferograms
<i>label_list</i>	List of strings containing names for the interferograms
<i>verbose</i>	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

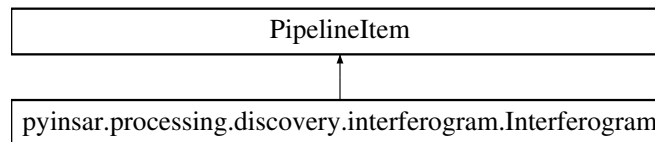
The documentation for this class was generated from the following file:

- `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

<i>str_description</i>	String describing item
<i>pairing</i>	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

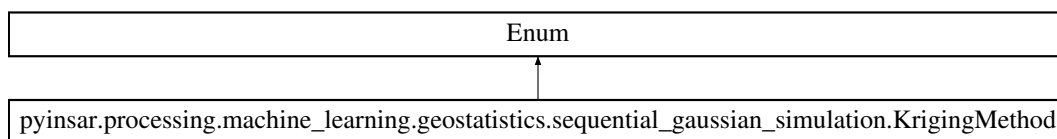
<i>obj_data</i>	Image wrapper containing SLC images
-----------------	-------------------------------------

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.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]
```

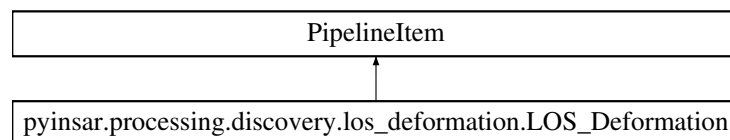
The documentation for this class was generated from the following file:

- 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:



6.15.1 Detailed Description

*** In Development ***

ap_paramList[]

def process(self, obj_data):

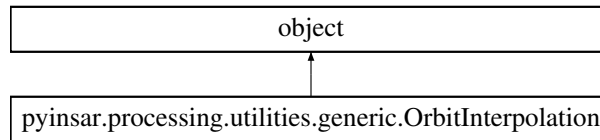
The documentation for this class was generated from the following file:

- 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

<i>orbit_data</i>	Orbit position data
<i>time_name</i>	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

<i>in_time</i>	Time of interest
<i>in_datetime</i>	Input is a datetime object (otherwise it's assumed its seconds from start date)
<i>interp</i>	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

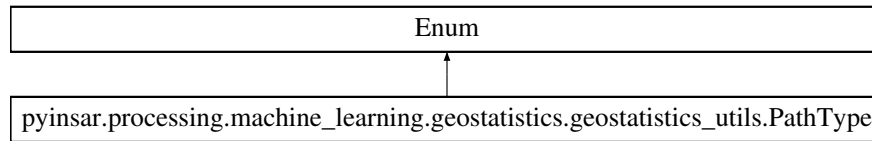
Starting date

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:



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]
```

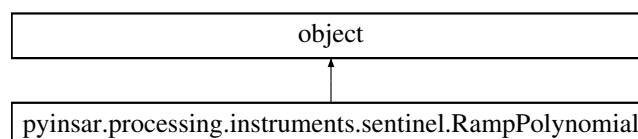
The documentation for this class was generated from the following file:

- 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

<i>t0</i>	Starting time
<i>coeff_list</i>	List of coefficients
<i>slant_range_time_interval</i>	Time between range samples
<i>slant_range_time</i>	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

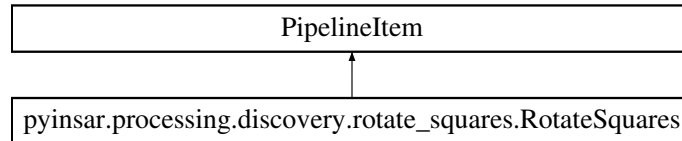
The documentation for this class was generated from the following file:

- [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

<i>str_description</i>	String describing class
<i>ap_paramList[SplineOrder]</i>	Spline order used in interpolation
<i>square_result_name</i>	Name of pipeline item that contains the Shapely squares
<i>angles</i>	Angles used when rotating squares
<i>clean</i>	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

<i>obj_data</i>	Image data wrapper
-----------------	--------------------

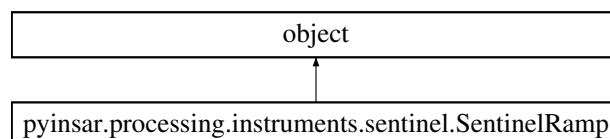
The documentation for this class was generated from the following file:

- processing/discovery/[rotate_squares.py](#)

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

Calculate 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 Sentinel 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

Calculate 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 Sentinel Ramp.

Parameters

<i>metadata</i>	ElemenTree containing the SLC metadata
<i>modulation</i>	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

<i>lines</i>	Index of lines
<i>samples</i>	Index of samples
<i>index</i>	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
```

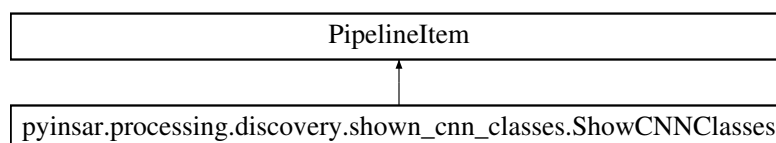
The documentation for this class was generated from the following file:

- processing/instruments/[sentinel.py](#)

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

Display CNN Classifications on segments of an image.

Inheritance diagram for 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

Display 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

<i>str_description</i>	String name of item
<i>class_name</i>	Name of classes
<i>colors</i>	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

<i>obj_data</i>	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
```

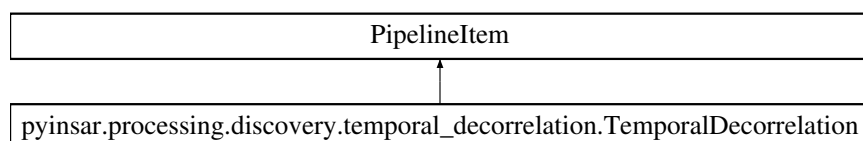
The documentation for this class was generated from the following file:

- 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:



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

<code>str_description</code>	String description of item
<code>ap_paramList[varyo_models]</code>	= Auto list of SGS models
<code>ap_paramList[varyo_sills]</code>	= Auto list of SGS sills
<code>ap_paramList[varyo_azimuth]</code>	= Auto param of SGS azimuth
<code>ap_paramList[varyo_ranges]</code>	= Auto list of SGS ranges
<code>ap_paramList[max_num_data]</code>	= Auto param of the max size of the neighborhood
<code>ap_paramList[decorrelation_mean]</code>	= Auto param of the decorrelation mean in the same units as the wavelength
<code>ap_paramList[decorrelation_std]</code>	= Auto param of decorrelation standard deviation in the same units as the wavelength
<code>grid_yx_spacing</code>	The y,x grid spacing
<code>wavelength</code>	Wavelength for converting to phase (from path length)
<code>seed</code>	Seed to use when generating noise
<code>save_noise</code>	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

<code>obj_data</code>	Image data wrapper
-----------------------	--------------------

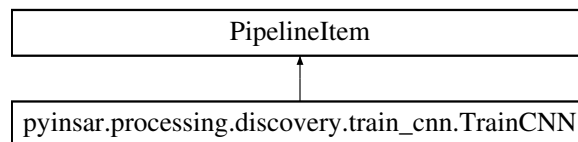
The documentation for this class was generated from the following file:

- 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

<i>str_description</i>	String describing item
<i>cnn_network_dir</i>	String containing the directory where the CNN is stored
<i>batch_size</i>	Batch size to use when training data
<i>config</i>	Dictionary 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

<i>obj_data</i>	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

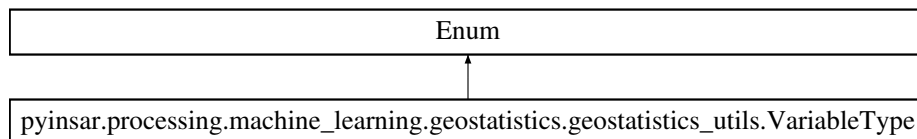
`pyinsar.processing.discovery.TrainCNN.config`

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`:



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]
```

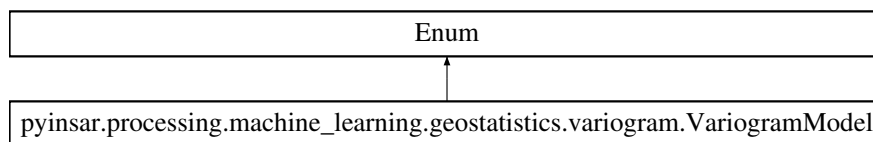
The documentation for this class was generated from the following file:

- [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`:



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]
```

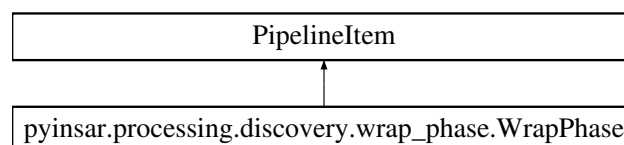
The documentation for this class was generated from the following file:

- 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

<i>obj_data</i>	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, file-
name=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 compatible 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↔
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.l1](#) (xi, eta, q, delta, nu, R, X, d_tild)
Okada's surface displacement.
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l2](#) (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l3](#) (xi, eta, q, delta, nu, R, y_tild, d_tild)
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l4](#) (xi, eta, q, delta, nu, R, d_tild)
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l5](#) (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.l1_int](#) (xi, eta, z, y, delta, c, d, q, R)
Okada's internal displacement.
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l2_int](#) (xi, eta, z, y, delta, c, d, q, R)
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l3_int](#) (xi, eta, z, y, delta, c, d, q, R)
- def [pyinsar.processing.deformation.elastic_halfspace.okada.l4_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 Development *** 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](#)
Display 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_subgeoraster](#) (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_latitude_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](#)
Calculate 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)
Calculate 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 thresholded.
- 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 ground 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

`__call__`
 `pyinsar::processing::instruments::sentinel::Ramp`↔
 Polynomial, [147](#)
 `pyinsar::processing::instruments::sentinel::Sentinel`↔
 Ramp, [150](#)
 `pyinsar::processing::utilities::generic::FindNearest`↔
 Pixel, [136](#)
 `pyinsar::processing::utilities::generic::OrbitInterpolation`,
 [145](#)
`__init__`
 `pyinsar::processing::data_fetcher::gdal::GDAL`↔
 DataFetcher, [140](#)
 `pyinsar::processing::data_fetcher::hdf_retriever::`↔
 DataRetriever, [123](#)
 `pyinsar::processing::data_fetcher::okada::Data`↔
 Fetcher, [122](#)
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::fault::Fault, [131](#)
 `pyinsar::processing::discovery::classify_cnn::`↔
 ClassifyCNN, [116](#)
 `pyinsar::processing::discovery::coherence::Coherence`,
 [118](#)
 `pyinsar::processing::discovery::coregister::Coregister`,
 [120](#)
 `pyinsar::processing::discovery::deburst::Deburst`, [128](#)
 `pyinsar::processing::discovery::deformation_to_`↔
 phase::DeformationToPhase, [129](#)
 `pyinsar::processing::discovery::flat_earth::FlatEarth`,
 [137](#)
 `pyinsar::processing::discovery::interferogram::`↔
 Interferogram, [141](#)
 `pyinsar::processing::discovery::rotate_squares::`↔
 RotateSquares, [148](#)
 `pyinsar::processing::discovery::shown_cnn_`↔
 classes::ShowCNNClasses, [152](#)
 `pyinsar::processing::discovery::temporal_decorrelation`↔
 ::TemporalDecorrelation, [154](#)
 `pyinsar::processing::discovery::train_cnn::TrainCNN`,
 [156](#)
 `pyinsar::processing::instruments::sentinel::Ramp`↔
 Polynomial, [147](#)
 `pyinsar::processing::instruments::sentinel::Sentinel`↔
 Ramp, [150](#)
 `pyinsar::processing::utilities::generic::FindNearest`↔
 Pixel, [135](#)
 `pyinsar::processing::utilities::generic::OrbitInterpolation`,
 [144](#)
 `pyinsar::processing::utilities::machine_learning::`↔
 DataRetriever, [125](#)
 `add_symmetric_border`
 `pyinsar::processing::geography::geomorphometry`,
 [59](#)
 `average_minmax_slices`
 `pyinsar::output::plot_raster`, [19](#)
 `batch_size`
 `pyinsar::processing::discovery::classify_cnn::`↔
 ClassifyCNN, [117](#)
 `pyinsar::processing::discovery::train_cnn::TrainCNN`,
 [157](#)
 `buildCNN`
 `pyinsar::processing::utilities::ann`, [90](#)
 CONTINUOUS
 `pyinsar::processing::machine_learning::geostatistics`↔
 ::geostatistics_utils::VariableType, [158](#)
 `calc_bounding_box`
 `pyinsar::processing::utilities::deformations`, [93](#)
 `pyinsar::processing::utilities::insar_simulator_utils`,
 [104](#)
 `cell_centroids`
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::fault::Fault, [132](#)
 `cell_length`
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::fault::Fault, [133](#)
 `cell_width`
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::fault::Fault, [133](#)
 `change_in_range_to_phase`
 `pyinsar::processing::utilities::insar_simulator_utils`,
 [104](#)
 `chinnerys_notation`
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::okada, [26](#)
 `chinnerys_notation_int`
 `pyinsar::processing::deformation::elastic_halfspace`↔
 ::okada, [26](#)
 `chunk_size`

pyinsar::processing::utilities::machine_learning::←
 DataRetriever, 126
 class_name
 pyinsar::processing::discovery::shown_cnn_←
 classes::ShowCNNClasses, 153
 classify
 pyinsar::processing::utilities::ann, 90
 cnn_network_dir
 pyinsar::processing::discovery::classify_cnn::←
 ClassifyCNN, 117
 pyinsar::processing::discovery::train_cnn::TrainCNN,
 157
 coherence
 pyinsar::processing::utilities::generic, 95
 colors
 pyinsar::processing::discovery::shown_cnn_←
 classes::ShowCNNClasses, 153
 compare_labels
 pyinsar::processing::discovery::classify_cnn::←
 ClassifyCNN, 117
 compute_averaged_cumulative_distribution_from_array
 pyinsar::processing::machine_learning::geostatistics_←
 ::sequential_gaussian_simulation, 77
 compute_axis_aligned_ellipse_range
 pyinsar::processing::machine_learning::geostatistics_←
 ::sequential_gaussian_simulation, 77
 compute_axis_aligned_neighborhood_shape
 pyinsar::processing::machine_learning::geostatistics_←
 ::sequential_gaussian_simulation, 77
 compute_closed_pipe_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::pipe, 42
 compute_continuous_distance
 pyinsar::processing::machine_learning::geostatistics_←
 ::direct_sampling, 68
 compute_delays
 pyinsar::processing::corrections::troposphere, 21
 compute_discrete_distance
 pyinsar::processing::machine_learning::geostatistics_←
 ::direct_sampling, 68
 compute_euclidean_distance
 pyinsar::processing::machine_learning::geostatistics_←
 ::sequential_gaussian_simulation, 77
 compute_experimental_variogram
 pyinsar::processing::machine_learning::geostatistics_←
 ::variogram, 84
 compute_fault_internal_displacement_type
 pyinsar::processing::deformation::elastic_halfspace_←
 ::okada, 26
 compute_gradient_at_cell
 pyinsar::processing::geography::geomorphometry,
 59
 compute_great_circle_distance_and_bearing
 pyinsar::processing::geography::geodesy, 55
 compute_horne_slope
 pyinsar::processing::geography::geomorphometry,
 59
 compute_lonlat_from_distance_bearing
 pyinsar::processing::geography::geodesy, 56
 compute_mogi_source_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::mogi, 24
 compute_neighborhood_lag_vectors
 pyinsar::processing::machine_learning::geostatistics_←
 ::direct_sampling, 68
 compute_neighborhood_template
 pyinsar::processing::machine_learning::geostatistics_←
 ::sequential_gaussian_simulation, 77
 compute_neighborhoods
 pyinsar::processing::machine_learning::geostatistics_←
 ::direct_sampling, 68
 compute_okada_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::okada, 27
 compute_okada_internal_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::okada, 27
 compute_open_pipe_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::pipe, 43
 compute_point_to_line_distance_for_array
 pyinsar::processing::geography::geodesy, 56
 compute_point_to_line_distance_on_ellipsoid
 pyinsar::processing::geography::geodesy, 56
 compute_range_variogram
 pyinsar::processing::machine_learning::geostatistics_←
 ::variogram, 84
 compute_uniform_disk_load_displacement
 pyinsar::processing::deformation::elastic_halfspace_←
 ::surface_load, 44
 compute_variogram
 pyinsar::processing::machine_learning::geostatistics_←
 ::variogram, 85
 compute_x_and_y_coordinates_maps
 pyinsar::processing::geography::coordinates, 49
 config
 pyinsar::processing::discovery::classify_cnn::←
 ClassifyCNN, 117
 pyinsar::processing::discovery::train_cnn::TrainCNN,
 157
 create_georaster_from_array
 pyinsar::output::export_georaster, 18
 create_product_xml
 pyinsar::processing::isce::input_file, 65
 create_topsApp_xml
 pyinsar::processing::isce::input_file, 65
 crop_array_from_center
 pyinsar::processing::utilities::insar_simulator_utils,

- 105
- crop_nans
 - pyinsar::processing::utilities::insar_simulator_utils, 105
- DISCRETE
 - pyinsar::processing::machine_learning::geostatistics↔::geostatistics_utils::VariableType, 158
- data_file_dict
 - pyinsar::processing::utilities::machine_learning::DataRetriever, 126
- data_import/import_georaster.py, 161
- data_import/import_raster.py, 161
- data_import/import_srcmod.py, 162
- data_import/import_utils.py, 162
- data_import/sentinel.py, 174
- data_import/uavsar.py, 162
- depth
 - pyinsar::processing::deformation::elastic_halfspace↔::fault::Fault, 133
- determine_deformation_bounding_box
 - pyinsar::processing::utilities::deformations, 94
 - pyinsar::processing::utilities::insar_simulator_utils, 105
- determine_x_y_bounds
 - pyinsar::processing::utilities::deformations, 94
 - pyinsar::processing::utilities::insar_simulator_utils, 106
- dip
 - pyinsar::processing::deformation::elastic_halfspace↔::fault::Fault, 133
- direct_vincenty_formula
 - pyinsar::processing::geography::geodesy, 57
- direct_vincenty_formula_for_array
 - pyinsar::processing::geography::geodesy, 57
- divide_into_squares
 - pyinsar::processing::utilities::machine_learning, 110
- download_file
 - pyinsar::data_import::import_utils, 14
- download_precise_orbits
 - pyinsar::data_import::sentinel, 15
- download_products
 - pyinsar::data_import::sentinel, 15
- dtype
 - pyinsar::processing::utilities::machine_learning::DataRetriever, 127
- EXPONENTIAL
 - pyinsar::processing::machine_learning::geostatistics↔::variogram::VariogramModel, 159
- ellipsoidal_earth_slant_ranges
 - pyinsar::processing::corrections::topography, 20
- exponential_variogram
 - pyinsar::processing::machine_learning::geostatistics↔::variogram, 85
- extract_subgeoarray
 - pyinsar::processing::geography::coordinates, 49
- f_x_dip
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 27
- f_x_strike
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 28
- f_x_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 28
- f_y_dip
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 28
- f_y_strike
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 28
- f_y_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 29
- f_z_dip
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 29
- f_z_strike
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 29
- f_z_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 29
- fA_1
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 30
- fA_1_dip
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 30
- fA_1_strike
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 30
- fA_1_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 30
- fA_2
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 31
- fA_2_dip
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 31
- fA_2_strike
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 31
- fA_2_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔::okada, 31

- fA_3
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 32
- fA_3_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 32
- fA_3_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 32
- fA_3_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 32
- fB_1
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 33
- fB_1_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 33
- fB_1_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 33
- fB_1_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 33
- fB_2
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 34
- fB_2_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 34
- fB_2_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 34
- fB_2_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 34
- fB_3
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 35
- fB_3_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 35
- fB_3_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 35
- fB_3_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 35
- fC_1
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 36
- fC_1_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 36
- fC_1_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 36
- fC_1_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 36
- fC_2
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 37
- fC_2_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 37
- fC_2_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 37
- fC_2_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 37
- fC_3
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 38
- fC_3_dip
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 38
- fC_3_strike
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 38
- fC_3_tensile
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 38
- find_closest_cell_in_training_image
 - pyinsar::processing::machine_learning::geostatistics↔
::direct_sampling, 69
- find_closest_time
 - pyinsar::processing::utilities::generic, 96
- find_data_asf
 - pyinsar::processing::utilities::generic, 96
- find_earthquake_pairs
 - pyinsar::processing::utilities::generic, 97
- find_overlapping_valid_lines
 - pyinsar::processing::instruments::sentinel, 61
- find_utm_area
 - pyinsar::processing::geography::coordinates, 50
- GAUSSIAN
 - pyinsar::processing::machine_learning::geostatistics↔
::variogram::VariogramModel, 159
- gaussian_variogram
 - pyinsar::processing::machine_learning::geostatistics↔
::variogram, 85
- generate_interferogram_from_deformation
 - pyinsar::processing::utilities::insar_simulator_utils,
106
- generate_minimum_ground_range_limits

- pyinsar::processing::utilities::machine_learning, 111
- generate_phase_samples
 - pyinsar::processing::utilities::machine_learning, 111
- generate_phase_samples_from_looks_and_ranges
 - pyinsar::processing::utilities::machine_learning, 112
- generateDeformation
 - pyinsar::processing::deformation::elastic_halfspace↔
::fault::Fault, 132
- generateMatplotlibRectangle
 - pyinsar::processing::utilities::generic, 97
- georaster_vertical_datum_shift
 - pyinsar::processing::geography::coordinates, 50
- get_data_to_data_matrix
 - pyinsar::processing::machine_learning::geostatistics↔
::sequential_gaussian_simulation, 78
- get_data_to_unknown_matrix
 - pyinsar::processing::machine_learning::geostatistics↔
::sequential_gaussian_simulation, 78
- get_georaster_array
 - pyinsar::data_import::import_georaster, 10
- get_georaster_extent
 - pyinsar::data_import::import_georaster, 10
- get_image_extents
 - pyinsar::processing::utilities::generic, 98
- get_images
 - pyinsar::processing::utilities::machine_learning::↔
DataRetriever, 126
- get_neighborhood
 - pyinsar::processing::machine_learning::geostatistics↔
::sequential_gaussian_simulation, 78
- get_num_images
 - pyinsar::processing::utilities::machine_learning::↔
DataRetriever, 126
- get_sentinel_extents
 - pyinsar::processing::instruments::sentinel, 61
- get_start_date
 - pyinsar::processing::utilities::generic::OrbitInterpolation, keypoints_align
145
- get_url_precise_orbit
 - pyinsar::data_import::sentinel, 16
- get_valid_lines
 - pyinsar::processing::instruments::sentinel, 62
- get_values_matrix
 - pyinsar::processing::machine_learning::geostatistics↔
::sequential_gaussian_simulation, 78
- I1
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 39
- I1_int
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 39
- I2
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 39
- I2_int
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 40
- I3
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 40
- I3_int
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 40
- I4
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 41
- I4_int
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 41
- I5
 - pyinsar::processing::deformation::elastic_halfspace↔
::okada, 41
- inverse_standard_normal_cdf
 - pyinsar::processing::machine_learning::geostatistics↔
::sequential_gaussian_simulation, 79
- inverse_vincenty_formula
 - pyinsar::processing::geography::geodesy, 57
- inverse_vincenty_formula_for_array
 - pyinsar::processing::geography::geodesy, 57
- is_any_equal
 - pyinsar::processing::machine_learning::geostatistics↔
::direct_sampling, 69
- is_any_nan
 - pyinsar::processing::machine_learning::geostatistics↔
::direct_sampling, 69
- k
 - pyinsar::processing::discovery::flat_earth::FlatEarth,
139
 - pyinsar::processing::utilities::generic, 98
- LINEAR
 - pyinsar::processing::machine_learning::geostatistics↔
::geostatistics_utils::PathType, 146
- label_list
 - pyinsar::processing::utilities::machine_learning::↔
DataRetriever, 127
- length
 - pyinsar::processing::deformation::elastic_halfspace↔
::fault::Fault, 133
- length_after_valid_window
 - pyinsar::processing::utilities::ann, 91
- map_2D_variogram
 - pyinsar::processing::machine_learning::geostatistics↔
::variogram, 85

- mask_deformation
 - pyinsar::processing::utilities::insar_simulator_utils, 107
- match_data
 - pyinsar::processing::utilities::generic, 99
- merge_secondary_data
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation, 79
- modulation
 - pyinsar::processing::instruments::sentinel::Sentinel↔Ramp, 151
- N
 - pyinsar::processing::corrections::troposphere, 21
- N_h
 - pyinsar::processing::corrections::troposphere, 22
- NUGGET
 - pyinsar::processing::machine_learning::geostatistics↔::variogram::VariogramModel, 159
- nogil
 - pyinsar::processing::geography::coordinates, 54
 - pyinsar::processing::machine_learning::geostatistics↔::geostatistics_utils, 75
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation, 81
 - pyinsar::processing::machine_learning::geostatistics↔::variogram, 87
- nopython
 - pyinsar::processing::geography::coordinates, 54
 - pyinsar::processing::geography::geodesy, 58
 - pyinsar::processing::geography::geomorphometry, 59
 - pyinsar::processing::machine_learning::geostatistics↔::direct_sampling, 71
 - pyinsar::processing::machine_learning::geostatistics↔::geostatistics_utils, 75
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation, 81
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation::Kriging↔Method, 142
 - pyinsar::processing::machine_learning::geostatistics↔::variogram, 87
- normal_score_transform
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation, 79
- normalize
 - pyinsar::processing::machine_learning::geostatistics↔::geostatistics_utils, 74
- nugget_variogram
 - pyinsar::processing::machine_learning::geostatistics↔::variogram, 85
- ORDINARY
 - pyinsar::processing::machine_learning::geostatistics↔::sequential_gaussian_simulation::Kriging↔Method, 143
- old_generate_interferogram_from_deformation
 - pyinsar::processing::utilities::insar_simulator_utils, 108
- open_gacos_tropospheric_delays
 - pyinsar::data_import::import_raster, 12
- open_georaster
 - pyinsar::data_import::import_georaster, 10
- open_sgems_file
 - pyinsar::data_import::import_raster, 12
- open_sgems_file_from_url
 - pyinsar::data_import::import_raster, 12
- output
 - pyinsar::processing::data_fetcher::gdal::GDAL_↔DataFetcher, 140
 - pyinsar::processing::data_fetcher::hdf_retriever::↔DataRetriever, 124
 - pyinsar::processing::data_fetcher::okada::Data↔Fetcher, 122
- output/export_georaster.py, 163
- output/plot_raster.py, 163
- pairing
 - pyinsar::processing::discovery::coherence::Coherence, 119
- parallel
 - pyinsar::processing::geography::coordinates, 55
- parse_satellite_data
 - pyinsar::data_import::sentinel, 16
- perturb
 - pyinsar::processing::data_fetcher::hdf_retriever::↔DataRetriever, 124
- phase_shift
 - pyinsar::processing::utilities::generic, 99
- phase_to_change_in_range
 - pyinsar::processing::utilities::insar_simulator_utils, 108
- plot_interactive_multiple_slicing
 - pyinsar::output::plot_raster, 19
- plot_interactive_slicing
 - pyinsar::output::plot_raster, 19
- poisson_ratio
 - pyinsar::processing::deformation::elastic_halfspace↔::fault::Fault, 133
- prepare_topsApps
 - pyinsar::processing::isce::input_file, 66
- prepare_training_image
 - pyinsar::processing::machine_learning::geostatistics↔::direct_sampling, 69
- print_georaster_info
 - pyinsar::data_import::import_georaster, 11
- process

- pyinsar::processing::discovery::classify_cnn::↵
ClassifyCNN, 116
- pyinsar::processing::discovery::coherence::Coherence, 119
- pyinsar::processing::discovery::coregister::Coregister, 121
- pyinsar::processing::discovery::deburst::Deburst, 128
- pyinsar::processing::discovery::deformation_to_↵
phase::DeformationToPhase, 130
- pyinsar::processing::discovery::flat_earth::FlatEarth, 137
- pyinsar::processing::discovery::interferogram::↵
Interferogram, 142
- pyinsar::processing::discovery::rotate_squares::↵
RotateSquares, 149
- pyinsar::processing::discovery::shown_cnn_↵
classes::ShowCNNClasses, 152
- pyinsar::processing::discovery::temporal_decorrelation↵
::TemporalDecorrelation, 155
- pyinsar::processing::discovery::train_cnn::TrainCNN, 156
- pyinsar::processing::discovery::wrap_phase::Wrap↵
Phase, 160
- processing/corrections/topography.py, 163
- processing/corrections/troposphere.py, 164
- processing/data_fetcher/gdal.py, 164
- processing/data_fetcher/hdf_retriever.py, 164
- processing/data_fetcher/okada.py, 167
- processing/deformation/elastic_halfspace/fault.py, 165
- processing/deformation/elastic_halfspace/mogi.py, 165
- processing/deformation/elastic_halfspace/okada.py, 165
- processing/deformation/elastic_halfspace/pipe.py, 167
- processing/deformation/elastic_halfspace/surface_load.↵
py, 168
- processing/discovery/classify_cnn.py, 168
- processing/discovery/coherence.py, 168
- processing/discovery/coregister.py, 168
- processing/discovery/deburst.py, 169
- processing/discovery/deformation_to_phase.py, 169
- processing/discovery/flat_earth.py, 169
- processing/discovery/interferogram.py, 170
- processing/discovery/los_deformation.py, 170
- processing/discovery/rotate_squares.py, 170
- processing/discovery/shown_cnn_classes.py, 171
- processing/discovery/temporal_decorrelation.py, 171
- processing/discovery/train_cnn.py, 171
- processing/discovery/wrap_phase.py, 171
- processing/geography/coordinates.py, 172
- processing/geography/geodesy.py, 173
- processing/geography/geomorphometry.py, 173
- processing/instruments/sentinel.py, 174
- processing/isce/input_file.py, 175
- processing/machine_learning/geostatistics/direct_↵
sampling.py, 175
- processing/machine_learning/geostatistics/geostatistics_↵
utils.py, 176
- processing/machine_learning/geostatistics/sequential_↵
gaussian_simulation.py, 177
- processing/machine_learning/geostatistics/variogram.py, 178
- processing/utilities/ann.py, 179
- processing/utilities/deformations.py, 180
- processing/utilities/generic.py, 180
- processing/utilities/insar_simulator_utils.py, 181
- processing/utilities/machine_learning.py, 182
- proj4StringToDictionary
pyinsar::processing::utilities::generic, 100
- project_insar_data
pyinsar::processing::utilities::generic, 100
- pyinsar, 9
- pyinsar.data_import, 9
- pyinsar.data_import.import_georaster, 9
- pyinsar.data_import.import_raster, 11
- pyinsar.data_import.import_srcmod, 13
- pyinsar.data_import.import_utils, 14
- pyinsar.data_import.sentinel, 15
- pyinsar.data_import.uavsar, 17
- pyinsar.output, 17
- pyinsar.output.export_georaster, 17
- pyinsar.output.plot_raster, 18
- pyinsar.processing, 20
- pyinsar.processing.corrections, 20
- pyinsar.processing.corrections.topography, 20
- pyinsar.processing.corrections.troposphere, 21
- pyinsar.processing.data_fetcher, 22
- pyinsar.processing.data_fetcher.gdal, 23
- pyinsar.processing.data_fetcher.gdal.GDAL_DataFetcher, 139
- pyinsar.processing.data_fetcher.hdf_retriever, 23
- pyinsar.processing.data_fetcher.hdf_retriever.Data↵
Retriever, 123
- pyinsar.processing.data_fetcher.okada, 23
- pyinsar.processing.data_fetcher.okada.DataFetcher, 121
- pyinsar.processing.deformation, 23
- pyinsar.processing.deformation.elastic_halfspace, 23
- pyinsar.processing.deformation.elastic_halfspace.fault, 24
- pyinsar.processing.deformation.elastic_halfspace.fault.↵
Fault, 130
- pyinsar.processing.deformation.elastic_halfspace.mogi, 24
- pyinsar.processing.deformation.elastic_halfspace.okada, 24
- pyinsar.processing.deformation.elastic_halfspace.pipe, 42
- pyinsar.processing.deformation.elastic_halfspace.↵
surface_load, 43
- pyinsar.processing.discovery, 44
- pyinsar.processing.discovery.classify_cnn, 45
- pyinsar.processing.discovery.ClassifyCNN, 115

- pyinsar.processing.discovery.Coherence, 118
- pyinsar.processing.discovery.coherence, 45
- pyinsar.processing.discovery.Coregister, 120
- pyinsar.processing.discovery.coregister, 45
- pyinsar.processing.discovery.deburst, 45
- pyinsar.processing.discovery.deburst.Deburst, 127
- pyinsar.processing.discovery.deformation_to_phase, 46
- pyinsar.processing.discovery.DeformationToPhase, 129
- pyinsar.processing.discovery.flat_earth, 46
- pyinsar.processing.discovery.FlatEarth, 136
- pyinsar.processing.discovery.interferogram, 46
- pyinsar.processing.discovery.interferogram.Interferogram, 141
- pyinsar.processing.discovery.LOS_Deformation, 143
- pyinsar.processing.discovery.los_deformation, 46
- pyinsar.processing.discovery.rotate_squares, 46
- pyinsar.processing.discovery.RotateSquares, 148
- pyinsar.processing.discovery.shown_cnn_classes, 47
- pyinsar.processing.discovery.shown_cnn_classes.Show↔
CNNClasses, 151
- pyinsar.processing.discovery.temporal_decorrelation, 47
- pyinsar.processing.discovery.TemporalDecorrelation, 153
- pyinsar.processing.discovery.train_cnn, 47
- pyinsar.processing.discovery.TrainCNN, 155
- pyinsar.processing.discovery.wrap_phase, 48
- pyinsar.processing.discovery.WrapPhase, 159
- pyinsar.processing.geography, 48
- pyinsar.processing.geography.coordinates, 48
- pyinsar.processing.geography.geodesy, 55
- pyinsar.processing.geography.geomorphometry, 58
- pyinsar.processing.instruments, 60
- pyinsar.processing.instruments.sentinel, 60
- pyinsar.processing.instruments.sentinel.RampPolynomial, 146
- pyinsar.processing.instruments.sentinel.SentinelRamp, 149
- pyinsar.processing.isce, 64
- pyinsar.processing.isce.input_file, 64
- pyinsar.processing.machine_learning, 66
- pyinsar.processing.machine_learning.geostatistics, 67
- pyinsar.processing.machine_learning.geostatistics.↔
direct_sampling, 67
- pyinsar.processing.machine_learning.geostatistics.↔
geostatistics_utils, 73
- pyinsar.processing.machine_learning.geostatistics.↔
geostatistics_utils.PathType, 146
- pyinsar.processing.machine_learning.geostatistics.↔
geostatistics_utils.VariableType, 157
- pyinsar.processing.machine_learning.geostatistics.↔
sequential_gaussian_simulation, 76
- pyinsar.processing.machine_learning.geostatistics.↔
sequential_gaussian_simulation.KrigingMethod, 142
- pyinsar.processing.machine_learning.geostatistics.↔
variogram, 83
- pyinsar.processing.machine_learning.geostatistics.↔
variogram.VariogramModel, 158
- pyinsar.processing.utilities, 89
- pyinsar.processing.utilities.ann, 90
- pyinsar.processing.utilities.deformations, 93
- pyinsar.processing.utilities.generic, 94
- pyinsar.processing.utilities.generic.FindNearestPixel, 135
- pyinsar.processing.utilities.generic.OrbitInterpolation, 144
- pyinsar.processing.utilities.insar_simulator_utils, 103
- pyinsar.processing.utilities.machine_learning, 110
- pyinsar.processing.utilities.machine_learning.Data↔
Retriever, 124
- pyinsar::data_import::import_georaster
get_georaster_array, 10
get_georaster_extent, 10
open_georaster, 10
print_georaster_info, 11
- pyinsar::data_import::import_raster
open_gacos_tropospheric_delays, 12
open_sgems_file, 12
open_sgems_file_from_url, 12
read_rsc_header_file, 13
- pyinsar::data_import::import_srcmod
read_srcmod_data, 13
- pyinsar::data_import::import_utils
download_file, 14
- pyinsar::data_import::sentinel
download_precise_orbits, 15
download_products, 15
get_url_precise_orbit, 16
parse_satellite_data, 16
- pyinsar::data_import::uavsar
read_uavsar_metadata, 17
- pyinsar::output::export_georaster
create_georaster_from_array, 18
- pyinsar::output::plot_raster
average_minmax_slices, 19
plot_interactive_multiple_slicing, 19
plot_interactive_slicing, 19
- pyinsar::processing::corrections::topography
ellipsoidal_earth_slant_ranges, 20
- pyinsar::processing::corrections::troposphere
compute_delays, 21
N, 21
N_h, 22
vapor_pressure, 22
- pyinsar::processing::data_fetcher::gdal::GDAL_Data↔
Fetcher
__init__, 140
output, 140
- pyinsar::processing::data_fetcher::hdf_retriever::Data↔
Retriever

- `__init__`, 123
- `output`, 124
- `perturb`, 124
- `pyinsar::processing::data_fetcher::okada::DataFetcher`
 - `__init__`, 122
 - `output`, 122
- `pyinsar::processing::deformation::elastic_halfspace↵`
 - `::fault::Fault`
 - `__init__`, 131
 - `cell_centroids`, 132
 - `cell_length`, 133
 - `cell_width`, 133
 - `depth`, 133
 - `dip`, 133
 - `generateDeformation`, 132
 - `length`, 133
 - `poisson_ratio`, 133
 - `strike`, 133
 - `unrotated_x`, 134
 - `unrotated_y`, 134
 - `width`, 134
 - `x_center`, 134
 - `y_center`, 134
- `pyinsar::processing::deformation::elastic_halfspace::mogi`
 - `compute_mogi_source_displacement`, 24
- `pyinsar::processing::deformation::elastic_halfspace↵`
 - `::okada`
 - `chinnerys_notation`, 26
 - `chinnerys_notation_int`, 26
 - `compute_fault_internal_displacement_type`, 26
 - `compute_okada_displacement`, 27
 - `compute_okada_internal_displacement`, 27
 - `f_x_dip`, 27
 - `f_x_strike`, 28
 - `f_x_tensile`, 28
 - `f_y_dip`, 28
 - `f_y_strike`, 28
 - `f_y_tensile`, 29
 - `f_z_dip`, 29
 - `f_z_strike`, 29
 - `f_z_tensile`, 29
 - `fA_1`, 30
 - `fA_1_dip`, 30
 - `fA_1_strike`, 30
 - `fA_1_tensile`, 30
 - `fA_2`, 31
 - `fA_2_dip`, 31
 - `fA_2_strike`, 31
 - `fA_2_tensile`, 31
 - `fA_3`, 32
 - `fA_3_dip`, 32
 - `fA_3_strike`, 32
 - `fA_3_tensile`, 32
 - `fB_1`, 33
 - `fB_1_dip`, 33
 - `fB_1_strike`, 33
 - `fB_1_tensile`, 33
 - `fB_2`, 34
 - `fB_2_dip`, 34
 - `fB_2_strike`, 34
 - `fB_2_tensile`, 34
 - `fB_3`, 35
 - `fB_3_dip`, 35
 - `fB_3_strike`, 35
 - `fB_3_tensile`, 35
 - `fC_1`, 36
 - `fC_1_dip`, 36
 - `fC_1_strike`, 36
 - `fC_1_tensile`, 36
 - `fC_2`, 37
 - `fC_2_dip`, 37
 - `fC_2_strike`, 37
 - `fC_2_tensile`, 37
 - `fC_3`, 38
 - `fC_3_dip`, 38
 - `fC_3_strike`, 38
 - `fC_3_tensile`, 38
 - `I1`, 39
 - `I1_int`, 39
 - `I2`, 39
 - `I2_int`, 40
 - `I3`, 40
 - `I3_int`, 40
 - `I4`, 41
 - `I4_int`, 41
 - `I5`, 41
 - `pyinsar::processing::deformation::elastic_halfspace::pipe`
 - `compute_closed_pipe_displacement`, 42
 - `compute_open_pipe_displacement`, 43
 - `pyinsar::processing::deformation::elastic_halfspace↵`
 - `::surface_load`
 - `compute_uniform_disk_load_displacement`, 44
 - `pyinsar::processing::discovery::classify_cnn::ClassifyCNN`
 - `__init__`, 116
 - `batch_size`, 117
 - `cnn_network_dir`, 117
 - `compare_labels`, 117
 - `config`, 117
 - `process`, 116
 - `size`, 117
 - `stride`, 117
 - `pyinsar::processing::discovery::coherence::Coherence`
 - `__init__`, 118
 - `pairing`, 119
 - `process`, 119
 - `use_progress_bar`, 119
 - `window`, 119
 - `pyinsar::processing::discovery::coregister::Coregister`

- `__init__`, 120
- `process`, 121
- `pyinsar::processing::discovery::deburst::Deburst`
 - `__init__`, 128
 - `process`, 128
- `pyinsar::processing::discovery::deformation_to_phase::DeformationToPhase`
 - `__init__`, 129
 - `process`, 130
- `pyinsar::processing::discovery::flat_earth::FlatEarth`
 - `__init__`, 137
 - `k`, 139
 - `process`, 137
- `pyinsar::processing::discovery::interferogram::Interferogram`
 - `__init__`, 141
 - `process`, 142
- `pyinsar::processing::discovery::rotate_squares`
 - `rotateSquare`, 47
- `pyinsar::processing::discovery::rotate_squares::RotateSquares`
 - `__init__`, 148
 - `process`, 149
- `pyinsar::processing::discovery::shown_cnn_classes::ShowCNNClasses`
 - `__init__`, 152
 - `class_name`, 153
 - `colors`, 153
 - `process`, 152
- `pyinsar::processing::discovery::temporal_decorrelation::TemporalDecorrelation`
 - `__init__`, 154
 - `process`, 155
- `pyinsar::processing::discovery::train_cnn::TrainCNN`
 - `__init__`, 156
 - `batch_size`, 157
 - `cnn_network_dir`, 157
 - `config`, 157
 - `process`, 156
- `pyinsar::processing::discovery::wrap_phase::WrapPhase`
 - `process`, 160
- `pyinsar::processing::geography::coordinates`
 - `compute_x_and_y_coordinates_maps`, 49
 - `extract_subgeoarray`, 49
 - `find_utm_area`, 50
 - `georaster_vertical_datum_shift`, 50
 - `nogil`, 54
 - `nopython`, 54
 - `parallel`, 55
 - `reproject_georaster`, 50
 - `reproject_point`, 51
 - `sample_2d_array`, 52
 - `sample_2d_multiarray`, 52
 - `sample_nd_array`, 52
 - `transform_to_geographic_coordinates`, 52
 - `transform_to_pixel_coordinates`, 53
 - `True`, 55
- `pyinsar::processing::geography::geodesy`
 - `compute_great_circle_distance_and_bearing`, 55
 - `compute_lonlat_from_distance_bearing`, 56
 - `compute_point_to_line_distance_for_array`, 56
 - `compute_point_to_line_distance_on_ellipsoid`, 56
 - `direct_vincenty_formula`, 57
 - `direct_vincenty_formula_for_array`, 57
 - `inverse_vincenty_formula`, 57
 - `inverse_vincenty_formula_for_array`, 57
 - `nopython`, 58
 - `update_lambda`, 58
- `pyinsar::processing::geography::geomorphometry`
 - `add_symmetric_border`, 59
 - `compute_gradient_at_cell`, 59
 - `compute_horne_slope`, 59
 - `nopython`, 59
- `pyinsar::processing::instruments::sentinel`
 - `find_overlapping_valid_lines`, 61
 - `get_sentinel_extents`, 61
 - `get_valid_lines`, 62
 - `read_geolocation`, 62
 - `retrieve_azimuth_time`, 62
 - `select_valid_lines`, 63
 - `transform_slc`, 63
 - `update_geolocation_lines`, 64
- `pyinsar::processing::instruments::sentinel::RampPolynomial`
 - `__call__`, 147
 - `__init__`, 147
- `pyinsar::processing::instruments::sentinel::SentinelRamp`
 - `__call__`, 150
 - `__init__`, 150
 - `modulation`, 151
- `pyinsar::processing::isce::input_file`
 - `create_product_xml`, 65
 - `create_topsApp_xml`, 65
 - `prepare_topsApps`, 66
- `pyinsar::processing::machine_learning::geostatistics::direct_sampling`
 - `compute_continuous_distance`, 68
 - `compute_discrete_distance`, 68
 - `compute_neighborhood_lag_vectors`, 68
 - `compute_neighborhoods`, 68
 - `find_closest_cell_in_training_image`, 69
 - `is_any_equal`, 69
 - `is_any_nan`, 69
 - `nopython`, 71
 - `prepare_training_image`, 69
 - `run_ds`, 70
 - `run_parallel_ds`, 70
 - `simulate_ds_realization`, 71

- pyinsar::processing::machine_learning::geostatistics↔
 - ::geostatistics_utils
 - nogil, 75
 - nopython, 75
 - normalize, 74
 - standardize, 74
 - True, 75
 - unflatten_index, 75
- pyinsar::processing::machine_learning::geostatistics↔
 - ::geostatistics_utils::PathType
 - LINEAR, 146
 - RANDOM, 146
- pyinsar::processing::machine_learning::geostatistics↔
 - ::geostatistics_utils::VariableType
 - CONTINUOUS, 158
 - DISCRETE, 158
- pyinsar::processing::machine_learning::geostatistics↔
 - ::sequential_gaussian_simulation
 - compute_averaged_cumulative_distribution_from_↔
 - array, 77
 - compute_axis_aligned_ellipse_range, 77
 - compute_axis_aligned_neighborhood_shape, 77
 - compute_euclidean_distance, 77
 - compute_neighborhood_template, 77
 - get_data_to_data_matrix, 78
 - get_data_to_unknown_matrix, 78
 - get_neighborhood, 78
 - get_values_matrix, 78
 - inverse_standard_normal_cdf, 79
 - merge_secondary_data, 79
 - nogil, 81
 - nopython, 81
 - normal_score_tranform, 79
 - run_parallel_sgs, 80
 - run_sgs, 80
 - simulate_sgs_realization, 80
 - solve_kriging_system, 81
 - True, 83
- pyinsar::processing::machine_learning::geostatistics↔
 - ::sequential_gaussian_simulation::Kriging↔
 - Method
 - nopython, 142
 - ORDINARY, 143
 - SIMPLE, 143
- pyinsar::processing::machine_learning::geostatistics↔
 - ::variogram
 - compute_experimental_variogram, 84
 - compute_range_variogram, 84
 - compute_variogram, 85
 - exponential_variogram, 85
 - gaussian_variogram, 85
 - map_2D_variogram, 85
 - nogil, 87
 - nopython, 87
 - nugget_variogram, 85
 - spherical_variogram, 86
 - True, 89
 - vectorized_exponential_variogram, 86
 - vectorized_gaussian_variogram, 86
 - vectorized_spherical_variogram, 87
- pyinsar::processing::machine_learning::geostatistics↔
 - ::variogram::VariogramModel
 - EXPONENTIAL, 159
 - GAUSSIAN, 159
 - NUGGET, 159
 - SPHERICAL, 159
- pyinsar::processing::utilities::ann
 - buildCNN, 90
 - classify, 90
 - length_after_valid_window, 91
 - restoreGraph, 91
 - shuffleTrainingData, 92
 - train, 92
- pyinsar::processing::utilities::deformations
 - calc_bounding_box, 93
 - determine_deformation_bounding_box, 94
 - determine_x_y_bounds, 94
- pyinsar::processing::utilities::generic
 - coherence, 95
 - find_closest_time, 96
 - find_data_asf, 96
 - find_earthquake_pairs, 97
 - generateMatplotlibRectangle, 97
 - get_image_extents, 98
 - keypoints_align, 98
 - match_data, 99
 - phase_shift, 99
 - proj4StringToDictionary, 100
 - project_insar_data, 100
 - rotate, 101
 - scale_image, 101
 - select_max_matched_data, 101
 - sorted_alphanumeric, 102
 - subarray_slice, 102
 - translate, 102
- pyinsar::processing::utilities::generic::FindNearestPixel
 - __call__, 136
 - __init__, 135
- pyinsar::processing::utilities::generic::OrbitInterpolation
 - __call__, 145
 - __init__, 144
 - get_start_date, 145
- pyinsar::processing::utilities::insar_simulator_utils
 - calc_bounding_box, 104
 - change_in_range_to_phase, 104
 - crop_array_from_center, 105
 - crop_nans, 105
 - determine_deformation_bounding_box, 105

- determine_x_y_bounds, 106
- generate_interferogram_from_deformation, 106
- mask_deformation, 107
- old_generate_interferogram_from_deformation, 108
- phase_to_change_in_range, 108
- retrieve_bounds, 109
- wrap, 109
- pyinsar::processing::utilities::machine_learning
 - divide_into_squares, 110
 - generate_minimum_ground_range_limits, 111
 - generate_phase_samples, 111
 - generate_phase_samples_from_looks_and_ranges, 112
 - rotate_image_list, 112
- pyinsar::processing::utilities::machine_learning::DataRetriever
 - __init__, 125
 - chunk_size, 126
 - data_file_dict, 126
 - dtype, 127
 - get_images, 126
 - get_num_images, 126
 - label_list, 127
 - size, 127
- RANDOM
 - pyinsar::processing::machine_learning::geostatistics::geostatistics_utils::PathType, 146
- read_geolocation
 - pyinsar::processing::instruments::sentinel, 62
- read_rsc_header_file
 - pyinsar::data_import::import_raster, 13
- read_srcmod_data
 - pyinsar::data_import::import_srcmod, 13
- read_uavsar_metadata
 - pyinsar::data_import::uavsar, 17
- reproject_georaster
 - pyinsar::processing::geography::coordinates, 50
- reproject_point
 - pyinsar::processing::geography::coordinates, 51
- restoreGraph
 - pyinsar::processing::utilities::ann, 91
- retrieve_azimuth_time
 - pyinsar::processing::instruments::sentinel, 62
- retrieve_bounds
 - pyinsar::processing::utilities::insar_simulator_utils, 109
- rotate
 - pyinsar::processing::utilities::generic, 101
- rotate_image_list
 - pyinsar::processing::utilities::machine_learning, 112
- rotateSquare
 - pyinsar::processing::discovery::rotate_squares, 47
- run_ds
 - pyinsar::processing::machine_learning::geostatistics::direct_sampling, 70
- run_parallel_ds
 - pyinsar::processing::machine_learning::geostatistics::direct_sampling, 70
- run_parallel_sgs
 - pyinsar::processing::machine_learning::geostatistics::sequential_gaussian_simulation, 80
- run_sgs
 - pyinsar::processing::machine_learning::geostatistics::sequential_gaussian_simulation, 80
- SIMPLE
 - pyinsar::processing::machine_learning::geostatistics::sequential_gaussian_simulation::KrigingMethod, 143
- SPHERICAL
 - pyinsar::processing::machine_learning::geostatistics::variogram::VariogramModel, 159
- sample_2d_array
 - pyinsar::processing::geography::coordinates, 52
- sample_2d_multiarray
 - pyinsar::processing::geography::coordinates, 52
- sample_nd_array
 - pyinsar::processing::geography::coordinates, 52
- scale_image
 - pyinsar::processing::utilities::generic, 101
- select_max_matched_data
 - pyinsar::processing::utilities::generic, 101
- select_valid_lines
 - pyinsar::processing::instruments::sentinel, 63
- shuffleTrainingData
 - pyinsar::processing::utilities::ann, 92
- simulate_ds_realization
 - pyinsar::processing::machine_learning::geostatistics::direct_sampling, 71
- simulate_sgs_realization
 - pyinsar::processing::machine_learning::geostatistics::sequential_gaussian_simulation, 80
- size
 - pyinsar::processing::discovery::classify_cnn::ClassifyCNN, 117
 - pyinsar::processing::utilities::machine_learning::DataRetriever, 127
- solve_kriging_system
 - pyinsar::processing::machine_learning::geostatistics::sequential_gaussian_simulation, 81
- sorted_alphanumeric
 - pyinsar::processing::utilities::generic, 102
- spherical_variogram
 - pyinsar::processing::machine_learning::geostatistics::variogram, 86
- standardize

pyinsar::processing::machine_learning::geostatistics↔
 ::geostatistics_utils, 74
 stride
 pyinsar::processing::discovery::classify_cnn↔
 ClassifyCNN, 117
 strike
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 133
 subarray_slice
 pyinsar::processing::utilities::generic, 102
 train
 pyinsar::processing::utilities::ann, 92
 transform_slc
 pyinsar::processing::instruments::sentinel, 63
 transform_to_geographic_coordinates
 pyinsar::processing::geography::coordinates, 52
 transform_to_pixel_coordinates
 pyinsar::processing::geography::coordinates, 53
 translate
 pyinsar::processing::utilities::generic, 102
 True
 pyinsar::processing::geography::coordinates, 55
 pyinsar::processing::machine_learning::geostatistics↔
 ::geostatistics_utils, 75
 pyinsar::processing::machine_learning::geostatistics↔
 ::sequential_gaussian_simulation, 83
 pyinsar::processing::machine_learning::geostatistics↔
 ::variogram, 89
 unflatten_index
 pyinsar::processing::machine_learning::geostatistics↔
 ::geostatistics_utils, 75
 unrotated_x
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 134
 unrotated_y
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 134
 update_geolocation_lines
 pyinsar::processing::instruments::sentinel, 64
 update_lambda
 pyinsar::processing::geography::geodesy, 58
 use_progress_bar
 pyinsar::processing::discovery::coherence::Coherence,
 119
 vapor_pressure
 pyinsar::processing::corrections::troposphere, 22
 vectorized_exponential_variogram
 pyinsar::processing::machine_learning::geostatistics↔
 ::variogram, 86
 vectorized_gaussian_variogram
 pyinsar::processing::machine_learning::geostatistics↔
 ::variogram, 86
 vectorized_spherical_variogram
 pyinsar::processing::machine_learning::geostatistics↔
 ::variogram, 87
 width
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 134
 window
 pyinsar::processing::discovery::coherence::Coherence,
 119
 wrap
 pyinsar::processing::utilities::insar_simulator_utils,
 109
 x_center
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 134
 y_center
 pyinsar::processing::deformation::elastic_halfspace↔
 ::fault::Fault, 134